DOI：10.6328/CIEE.2016.2.05

NewSQL, New database era

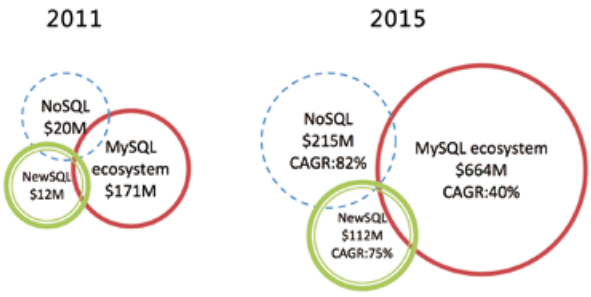
古永忠
資策會創研所

一、 Introduction

在資料庫系統技術發展已超過 30 年之後，“The Traditional RDBMS Wisdom is All Wrong”[1]，Postgres 的創辦人及 MIT 教授的 Michael Stonebreaker 卻做了這樣的表示。

Big data 是近年來資料庫技術開始改變的契機，因為資料型態與應用的改變，傳統資料庫技術開始遭遇挑戰，穩固的基石也隨之動搖，也迫使各界開始重新定義與發展新的資料庫系統技術。以代表性而論，從 SQL 類的關連式資料庫，到革新的 NoSQL 類資料庫，至新興的 NewSQL 類資料庫，它們代表的並非是另一套資訊系統分類，而是引領資料庫系統思維的創新。

由圖一之市場分析可知，NewSQL 並非是一統江湖的概念，事實上，無論 SQL 類資料庫或 NoSQL 類資料庫，都有其適用的市場，彼此之間關係為互補，而非競爭。這與過去僅有 SQL 類資料庫的市場組成已完全不同。而 NewSQL 的定義，由於百家爭鳴，本文所採用的是較為簡潔的概念，由 451group[3] 所提出的：SQL-like 語法介面、NoSQL 所提倡的延展性、應用專屬資料設計。



圖一　MySQL, NoSQL, NewSQL revenue[2]

簡而言之，NewSQL 類資料庫系統企圖在 SQL 與 NoSQL 之間取得某種平衡點，而其取捨即歸於特定應用專屬之資料處理需求。資料庫系統處理的是人與資料的關聯，而資料庫產品市場也已走向多元性發展，傳統上所謂「最好的資料庫系統」已不存在，而是以使用者為導向，建立資料應用思維，方能決定最適合的資料庫系統。

二、 SQL and Relational database

首先，必須要把 SQL（Structured query language[4]）與關聯式資料庫（Relational database）作區分。雖然 SQL 是由關聯式資料庫的概念衍生而來，但不代表 SQL 只能應用在關聯式資料庫之上。SQL 之意義如同這世界上的各種自然語言，它經歷長時間的考驗與調整，於學理及實用上，都能泛用性地表達各種資料處理的方式。而關聯式資料庫系統能以更接近人的方式來面對資料，SQL 即是其中一項重要的設計。

就目前市場的實際情況來看，關聯式資料庫自然都採用 SQL 語法，但也有一些的 NoSQL 類的資料庫設計是 SQL-like 的語法，如 Pig[5]、Hive[6]、InfluxDB[7]，它並非完整對應 ANSI SQL，但其功能性無非是希望提升使用的親切度。是故，雖然資料庫系統角色上為後端系統，但其人機介面仍是必須要被重視的一環。

資料在被處理成為「資訊」之前，是無法被有效運用的。所以資料庫必須要俱備「運算」的能力，而不僅僅是儲存資料而已。而在現今所謂 Big data 的時代，in-database processing 的概念也應韻而生。其起因為資料的遷移已大量侵蝕資料處理的時間，所以盡可能讓計算在資料庫內發生，以避免不必要的存取損耗。NoSQL 類的資料庫通常針對特定的資料處理需求，而設計了特定的計算與儲存方式，但終究無法在資料庫內處理千奇百怪的商用邏輯。而「in-database processing」在關聯式資料庫中已行之有年，如 Oracle[8] 的 PL/SQL[9]、PostgreSQL[10] 的 PL/pgSQL[11]、PL/python[12] 等程式語言，它們能讓開發者把資料處理邏輯嵌入資料庫中，讓資料庫能高彈性地同時儲存與運算資料。

如同前文所引用 Michael Stonebreaker 的話語，SQL 類資料庫的概念在許多人的腦海中已是既定的知識，但它也限制了新型態資料處理的發展，所以產生了 NoSQL 類的資料庫系統。然而，正處於資料世代的我們，要做的並非全然拋棄這一切，而是在傳統的知識基礎上尋求改變之道，截長補短，主角是人與資料，缺一不可。

然而，SQL 的設計目標為人性化操作，但也伴隨著許多的代價，例如 SQL Parser 執行效率不佳、非結構化資料型態支援不足等。NewSQL 資料庫嘗試在這些缺陷中進行取捨與調整，讓相容性與效能取得良好的平衡點。下段介紹的 NoSQL 類資料庫則有不同的觀念與作法，但無須分高下，而是作為多元觀點的思考，找到讓使用者與資料都能溶入應用服務的最佳切入點。

三、 NoSQL is not only SQL

若由 Oracle 於 1978 年上市的第一個資料庫系統起算至今，資料庫技術的發展已接近 40 年。資料庫領域呈現高度成熟與穩定的態勢，直到 1998 年 NoSQL 這個名詞的出現，資料庫的世界才開始產生重大的改變。其關鍵問題在於，傳統關聯式資料庫所堅持的資料一致性（Consistency），在新型態的資料需求產生之後，這樣的嚴謹反而成為一種阻礙。

CAP theorem 是由 Eric Brewer[13] 所提出的理論，意指在分散式系統中，Consistency、Availability、Partition tolerance 三種訴求只能保證其二。在 Big data 時代來臨後，資料量遠大過單一主機的處理上限，分散式系統成為顯學，此理論的限制立即被突顯。由於可用性（Availability）是各種系統公認所必須要求的基本，故多數系統在一致性（Consistency）與分散容錯性（Partition tolerance）作出選擇。傳統的關聯式資料庫系統以一致性為優先，

所以分散式的關聯式資料庫系統總是有明顯的分散瓶頸；而 NoSQL 類資料庫決定挑戰以 Partition tolerance 為優先的系統設計方式，成功地開創了新的資料庫領域。

NoSQL 做了更多資料導向的選擇，雖然使用者可能無法安心確定何時會達到同步（Eventual consistency），但可以讓資料有更快速的處理；語法上可能彈性少了，但可以讓執行流程更有效率。

資料多樣性（Data variety）是現今資料型態的現況，「你永遠不知道下一個資料型態是什麼？」根據 db-engines.com[14] 所統計的數量，已達到約 300 個資料庫產品數量。各種資料庫都有其專注的資料問題，所以到現在這個時代，選擇資料庫本身也成為一個重要的課題了。例如 Key-value 資料庫即 NoSQL 由繁趨簡的先驅，Memcached[15] 是目前最具代表性的產品。這類的資料庫講究效率，也易於分散式處理，極簡化資料存取，適用於快取類的應用。

Graph database 是近年來成長快速的資料庫，代表性的產品為 Neo4j[16]，其針對的議題為資料之間的關聯性，因社群網路的發展而開始受到重視，而 Graph 問題的運算複雜度又正巧是傳統關聯式資料庫最痛的問題點之一，促使 Graph database 的壯大。Graph database 有效地提供 data mining 新的契機，不論深度和廣度的資料搜尋，都有長足的進步。

Text search 資料庫也因為輿情分析的需求而興起。全文檢索技術雖然發展較為成熟，但一直未發展成為獨立的資料庫系統。目前以 Elasticsearch[17] 最為熱門，它簡化了大量的文字資料索引的統整工作，並輔以管理一般數字及時間等相關資訊。對於欲建立私有文字資料庫的應用服務，有巨大的貢獻。目前 wikimedia 即使用 Elasticasearch 作為其文字檢索資料庫 [18]。

IOT（Internet of things）的需求引發了 Time series database 發展，其訴求在於快速計算特定時間區間的統計值，並以統計模型進行未來趨勢評估。常見的應用於大量的感應器收集環境資料，即時性評估如機器壽命、結構風險、醫療警示等應用。其以時間戳記為核心，處理不同時間顆粒及區間的統計計算，也提供連續值點的填補與預估，以提供決策的需求。

四、NewSQL is in

講究泛用性的 SQL 類資料庫及重視專用性的 NoSQL 類資料庫，剛好追求的是兩個極端，而 NewSQL 則提倡平衡，平衡點則為應用服務的需要。資料庫系統的多元化發展，來自於資訊應用需求更為廣泛，傳統資料庫觀念無法一體適用。所以雖然 NewSQL 定義上並未限定何種應用，但目前市場上 NewSQL 類產品均以資料分析作為設計的主要標的。

SQL 資料庫的出現，是協助「人」簡單處理資料問題；NoSQL 資料庫則是由「資料」為出發點，尋求資料專屬的處理系統。這其中還有一個缺口，就是「應用」，而這是 NewSQL 類資料庫所訴求的部份。

以目前多數 NewSQL 資料庫設定為分析用資料庫為例，分析人員不盡然是資訊技術醇熟的操作人員，所以多納入較親切的 SQL 語法；交易安全及備援可能就不是優先的考量，所以多採用 In-memory 的技術加快執行效率，並調整交易安全等級，減少不必要的延遲；在 Big data 的潮流下，NoSQL 資料庫於分散式儲存的技術是必須要納入的。

In-memory relational database 是 NewSQL 中成長快速的一個類別，如 VoltDB[19]、MemSQL[20] 等產品都有相當優秀的表現，重點是重新針對 In-memory 的特點進行系統設計，在確保資料一致性的情況下，大幅提升執行效率，也具備一定程度的分散能力。NewSQL 講究的不是極端的 CAP theorem，而是 CAP 三者的平衡點。In-memory relational database 可以讓使用者當成傳統的關聯式資料庫來使用，並善用記憶體的速度，來突破應用所需的效能瓶頸。

Multi-model database 是另一個發展方向，它結合了不同類型資料庫的特點於一身。現時多數實際的應用，由於資料型態變化大，都可能必須整合兩種或以上的資料庫系統，以符合其應用發展需求。故 Multi-model database 期待能在單一資料庫，在接近專用資料庫效率的前提下，解決多樣化資料的問題，以減少管理及開發的複雜度。以 FoundationDB[21] 為例，目前已被 Apple 公司所併購 [22]，主要看上的就是 Multi-model database 的高使用彈性。同樣使用 SQL 介面，但可以同時擁有 Key-value 的快速，以及 ACID 的安心。

SQL 與 NoSQL 的串連也是重頭戲，Apache Trafodion[23] 專案是近來具代表性的例子，Transactional SQL-on-Hadoop Database。目標十分明確，就是要能兼顧 SQL 所提供的親切介面和資料交易安全，以及 Hadoop 平台的分散處理優勢。這個案例突顯了資料庫系統在操作介面與資料處理功能實際上是同等重要的。

五、Database ecosystem

在資訊系統領域之中，空有系統功能不一定能有所貢獻。目前凡俱有一定使用量的軟體，其背後都有著強大的軟體生態系在支撐著，其必然包含了技術面、應用面、工具面、及商業面的支援。換句話說，軟體是由人來使用的，它必須要滿足各種使用角度的人，才能夠持續發展。然而，沒有任何一套軟體是完美的，所以需要一個軟體生態系來互利共生。老牌的 PostgreSQL，歷史悠久，是典型的關聯式資料庫系統，近年來積極轉型，卻仍不捨既有穩定發展的特色，故以其為例說明其生態系的發展應具有代表性。

PostgreSQL 始於西元 1996 年，至今約 20 年，並一直以 Open source 的方式發展。開發團隊有良好的制度，提供長期的營運，並且使用者眾多，才能長期位居熱門系統之中，以 db-engines.com 所統計的排名而言，均在前 5 名之譜。尚不論功能是否強大，至少其滿足了開發者與使用者多數的需求，才能在時代的洪流中存續。資料庫系統可說是整個應用服務的黑手，它絕對不會是最酷炫的那個模

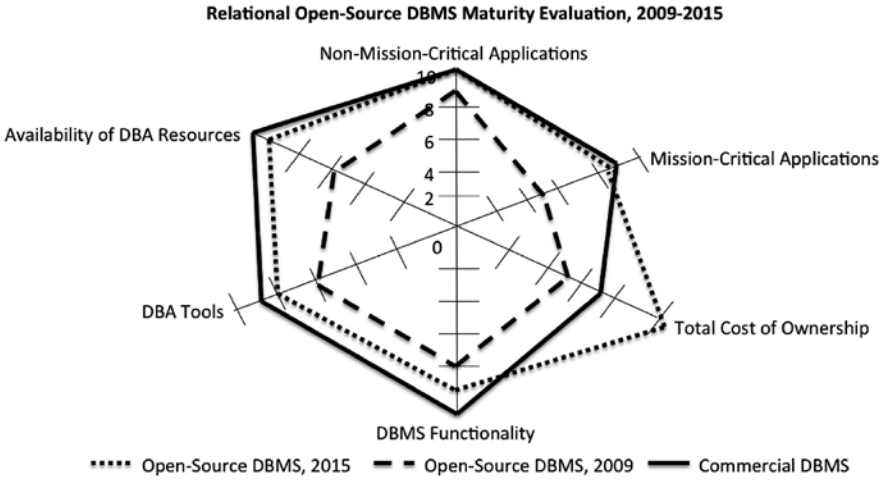
樣，卻必須肩負著資料穩定存取的信任感。PostgreSQL 長期專注於關聯式資料庫的核心發展，依賴展現其最尖端的應用服務眾多。

軟體是必須要被「人」所使用才有意義，良好的技術是必要，但並非唯一。長期使用之下，支援工具的完整度也是相當重要的事。任何系統都無法避免完全不產生錯誤，日常的監控系統就有其必要性，PostgreSQL 在知名監控系統 Zabbix 上有 pg_monz[25]，也有開源的 Open PostgreSQL Monitoring（opm.io）專案 [26]；圖型化資料庫管理介面常見的 pgAdmin[27]，也有 Web-based PostgreSQL Studio[28]；還有輕量化 Connection pool 的 pgBouncer[29]；支援分散式資料庫的 pgPool[30]……等等。這些工具能讓資料庫系統更接近人與應用的需求，簡化工具流程，才有利於持續使用。

圖二為知名市場分析商 Gartner 對資料庫產業成熟度的調查比較，可見商業化支援一直是開源軟體無法進入主流市場的關鍵，其相關的問題不一定在於軟體本身，而是在於人員的技術支援，終究必須要能夠解決人的問題才能佔有其市場。這部份如 Oracle database 或 Microsoft SQL Server[31]，均是熟稔商業經營的領導廠商，凡舉軟體問題、商業合約問題、人力資源問題，都是商業化必須面對的。PostgreSQL 近年來才開始有廠商來補強這一塊， 如 EnterpriseDB[32]、CrystalDB[33] 等。而台灣的軟體人才輸出，雖然在技術面可達世界水準，卻在產業面的商業支援一直都相對貧乏，以致於在資料庫系統軟體，呈現大量入超現象。

六、iServDB is a kind of NewSQL database system

本資料庫團隊（資策會創研所）自民國 101 起，開始執行經濟部技術處所支持的「工業基礎技術研究計畫」[34] 負責其「分散式

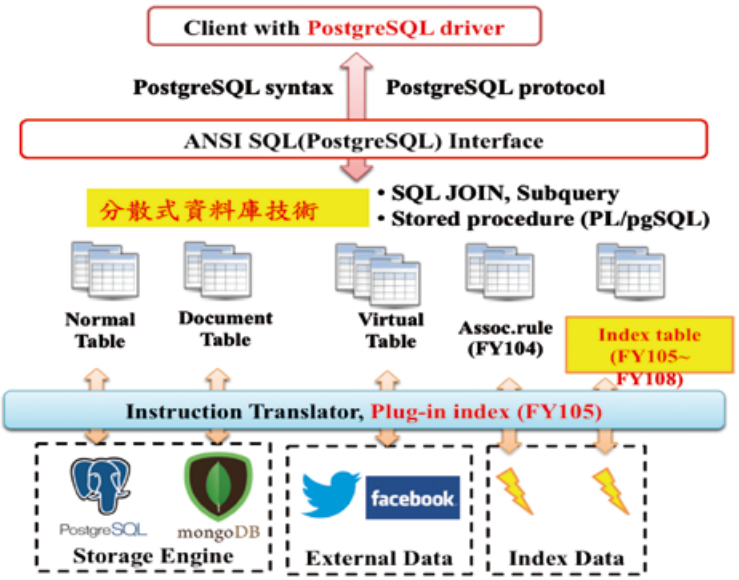


圖二　開源資料庫成熟度進度圖 [24]

資料庫技術」之研發，目前使用「iServDB」（http://iservdb.cloudopenlab.org.tw/）[35] 作為產品名稱對外推廣。iServDB 之產品目標以適應國內產業需求為主，並整合國內產學技術供應鏈，形成自主技術良好的生態發展。

iServDB 技術上的期待是能兼顧穩定發展與高度彈性，所以經綜合評估後，採 PostgreSQL 作為主要的技術載具，一則為開源發展的 PostgreSQL 有利於學習世界領先之資料庫技術；另一則為在國內資料庫產業相對疲弱之際，乘其之勢，縮短資料庫產業生態系與國際市場之差距。於穩定發展的面向，iServDB 強化自動佈署、即時監控、相互備援、效率介面等與營運管理有關的項目；而高度彈性則研發了分散式資料庫技術，增強資料庫動態配置的能力，並延伸 PostgreSQL foreign data wrapper 機制，連結 NoSQL 類資料庫，如 MongoDB[36]、Elasticsearch 等，這與 Multi-model database 的設計類似，使多樣化的資料處理，能在單一 SQL 介面、單一資料庫系統中輕鬆完成。

如前文所述，現今的資料庫系統已無法以一擋百，必須選擇好所面對的資料市場，而 iServDB 選擇的是資料分析的應用。再進一步來說，iServDB 所設定的使用者為資料分析人員。由於資料分析行為的特性，它必須針對每一個案例進行微調其查詢行為，與傳統資料探勘行為常見的批次運算模式並不相同。故 iServDB 企圖發展一套能與資料分析人員快速互動的 Interactive system，所以於技術上就必須擁有 Bounded response time 的系統。類似相關的技術目前由 U.C. Berkeley 與 M.I.T. 的聯合實驗室研發的 BlinkDB[37]，即進行此特性系統的領導性研究，專注於數值性資料的研究。

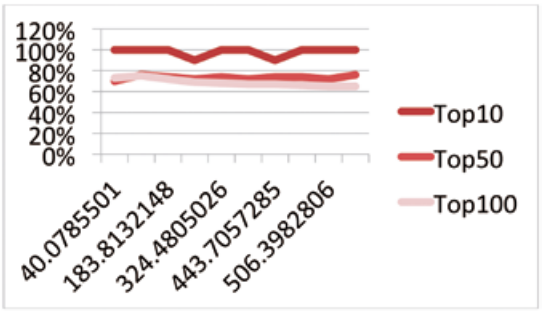


圖三　分散式資料庫技術系統架構圖

iServDB 基於工業基礎技術研究計畫所要求的高技術挑戰性，挑戰過去只應用在數值上的抽樣統計方法，意即我們可能在人力物力無法完全計算之時，可利用已計算的結果，推估最終可能的數值範圍，常見於民意調查活動。與民意調查的情況相同，現在我們想要觀測的資料量已大到無法完全計算的程度了，單純數值統計是如此，還有更多複雜的非結構化資料在等著我們。

以 iServDB 所研發的高精確型 Association rule 系統方法為例，這類的資料分析方法不同於數值統計方法單純，典型的困擾即是資料運算時間不容易估計，使得分析決策的時效性也難以確保。而 Bounded response time 的研究即是限制其回應時間，不論運算量有多大，重點在於可預估其正確率，如此即可在決策的時效性取得平衡點。依本研究實驗結果，1 分鐘內即可達到 7 成的正確率，並且每段時間都能不斷更新結果，而不一定要忍受無法估計的等待。

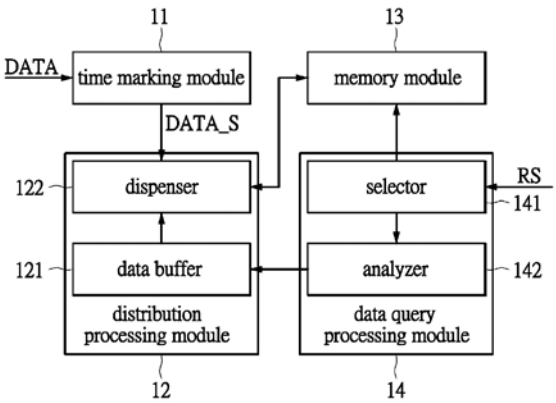
圖四橫軸為執行時間，縱軸為執行結果與傳統結果的比較後的準確率。由於分析應用通常只在乎排序最前端的結果，並不追求完全正確的清單（可能高達數千項，依 support 設定而有所不同），所以區分為 Top10（前 10 項結果比較）、Top50、Top100 等項目呈現。此研發實驗主要目標為突顯出，資料的計算在回應時間上是可以有彈性的，而典型的 Association rule 運算僅有「完成」與「未完成」兩極化的結果，它可以和既有完



圖四　實驗結果摘要圖

整運算的演算法互補，重點在於符合應用的需求，讓資料需求來領導資料運算。

另外，iServDB 也投入 Time series 運算的研發，其相對應的應用為 IOT，此類問題資料內容單純卻量大，但其關鍵在於大量的時序資料查詢，不同時間顆粒的重覆查詢也不斷改變，時常造成大量負載。iServDB 研發可分散式且輕量化的時間序列索引機制，能分散處理，並大量縮減資料運算量，以 Bounded response time 的理念，確保隨時可回應的狀態，並逐步趨近理想值。此研發佈局未來，已申請台灣專利 [38] 及美國專利 [39] 並且獲證。



圖五　Time series 專利示意圖

iServDB 瞭解到世界脈動是瞬息萬變，資料本身也是越來越多樣化，所以除了 SQL 介面仍然保留之外，其他部份都採高彈性化設計，模組化可隨意組合。如 NewSQL 的思維一般，依應用選擇資料庫的功能，既保留 SQL 的介面也享受 NoSQL 的高效能。

七、The future of data

現在已確確實實是資料世代，如同街頭的監視攝影機，大量的資料收集裝置已遍佈我們的生活四週，一舉一動都形成資料被記錄下來；而每個人也都想要從這些資料中洞察世界的趨勢，運算越快越好，越廣越好。現在不論是物流、銷售、運動、休閒、機械、交通……各行各業都追著資料跑，期待能從資料中得到未來的模樣。不過要做到這些之前，我們都必須要更瞭解資料庫技術才行。

如同先前提到的 CAP theorem，這三項都是人們期待資料庫系統能夠完美達成的，但卻是相互矛盾的性質。資料庫系統的設計，實際上就是一連串取捨的過程。資料的處理在多樣化的訴求下，無法一體適用，選擇適當的資料庫系統便成為現今資料應用者的課題。

以技術發展人員來說，選擇資料系統的生態系是重要的。每個現存的資料庫系統都有其生存之道，投入適當的生態系，隨著更為順暢的資料流，技術有利於擴散，就能讓更多使用者看見你的技術。

資料庫系統從穩定發展到百家爭鳴；從高度泛用的 SQL 資料庫，到講究專用的 NoSQL 資料庫；現在再到向應用靠攏的 NewSQL 資料庫，未來必定還會有更多創新多變的資料庫系統產生。始終不變的是，人們把資料視為重要的價值，而資料庫系統則是最好的橋樑。

【參考資料】

[1] Michael Stonebreaker (May 30, 2013). One Size Fits None - (Everything You Learned in Your DBMS Class is Wrong). http://slideshot.epfl.ch/play/suri_stonebraker

[2] The 451 group (May 22, 2012). 451 Research delivers market sizing estimates for NoSQL, NewSQL and MySQL ecosystem. https://blogs.the451group.com/information_management/2012/05/22/mysql-nosql-newsql/

[3] Ivan Glushkov (Feb 13, 2015). NewSQL Overview. http://www.slideshare.net/IvanGlushkov/newsql-overview

[4] International Organization for Standardization: “ISO/IEC 9075-14:2008”

[5] Apache Pig. https://pig.apache.org/

[6] Apache HIVE. https://hive.apache.org/

[7] InfluxDB. https://influxdata.com/

[8] Oracle Database. https://www.oracle.com/database/index.html

[9] Oracle Database PL/SQL. http://www.oracle.com/technetwork/database/features/plsql/index.html

[10] PostgreSQL. http://www.postgresql.org/

[11] PL/pgSQL – SQL Procedural Language. http://www.postgresql.org/docs/current/static/plpgsql.html

[12] PL/Python – Python Procedural Language. http://www.postgresql.org/docs/current/static/plpython.html

[13] “Brewer’s CAP Theorem”, julianbrowne.com,

Retrieved 02-Mar-2010

[14] DB-Engines. http://db-engines.com/

[15] memcached – a distributed memory object caching system. https://memcached.org/

[16] Neo4j: The World's Leading Graph Database. http://neo4j.com/

[17] Elastic · Revealing Insights from Data (Formerly Elasticsearch). https://www.elastic.co/

[18] Chad Horohoe, Wikimedia Foundation (January 6, 2014). Wikimedia moving to Elasticsearch. http://blog.wikimedia.org/2014/01/06/wikimedia-moving-to-elasticsearch/

[19] VoltDB: In-Memory Operational Database, SQL and Scale-Out. https://voltdb.com/

[20] MemSQL: The Fastest In-Memory Database. http://www.memsql.com/

[21] FoundationDB. https://foundationdb.com/

[22] Matthew Panzarino. Apple Acquires Durable Database Company FoundationDB. http://techcrunch.com/2015/03/24/apple-acquires-durable-database-company-foundationdb/

[23] Apache Trafodion. http://trafodion.apache.org/

[24] Gartner. The State of Open-Source RDBMSs, 2015. https://www.gartner.com/doc/3033819/state-opensource-rdbmss-

[25] PostgreSQL monitoring template for Zabbix (pg_monz). http://pg-monz.github.io/pg_monz/index-en.html

[26] Open PostgreSQL Monitoring. http://opm.io/

[27] pgAdmin: PostgreSQL administration and management tools. http://www.pgadmin.org/

[28] PostgreSQL Studio. http://www.postgresqlstudio.org/

[29] PgBouncer - lightweight connection pooler for PostgreSQL. https://pgbouncer.github.io/

[30] Pgpool Wiki. http://www.pgpool.net/

[31] SQL Server 2014 | Microsoft. https://www.microsoft.com/zh-tw/server-cloud/products/sql-server/

[32] EnterpriseDB | The Postgres Database Company. http://www.enterprisedb.com/

[33] CrystalDB Inc. http://www.crystaldb.com/

[34] 深耕工業基礎技術發展，https://www.moea.gov.tw/MNS/doit/content/Content.aspx?menu_id=13451

[35] iServDB. http://iservdb.cloudopenlab.org.tw

[36] MongoDB for GIANT Ideas | MongoDB. https://www.mongodb.org/

[37] BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data. http://blinkdb.org/

[38] 古永忠、蔡宗融、陳立群，資料處理裝置及資料處理方法，中華民國發明第 I526966 號。

[39] Yung-Chung Ku, Jonathan Tsai, Lee Chung Chen. Data processor and a data processing method. U.S. Patent No.: US 9,262,466 B2