

คำนำ

ขณะนี้เรามีฐานข้อมูลให้เลือกใช้เป็นร้อยเป็นพัน แล้วทางเลือกไหนที่จะเหมาะกับแอปพลิเคชันของเรา? คำตอบแบบสั้นคือ “ก็ขึ้นอยู่กับงาน” ส่วนคำตอบแบบเจาะลึกก็คือ ... หนังสือเล่มนี้

เทคโนโลยีการจัดเก็บและประมวลผลข้อมูลแต่ละแบบล้วนมี **สิ่งที่ต้องแลกเปลี่ยน (Trade-off)** ที่ต่างกัน และไม่มีแนวทางใดที่ยอดเยี่ยมที่สุดสำหรับทุกสถานการณ์ ระบบที่สมบูรณ์แบบสำหรับแอปหนึ่งอาจกลายเป็นฝันร้ายของอีกแอปก็ได้ หนังสือเล่มนี้คือคู่มือท่องไปในอาณาจักรของระบบข้อมูลทั้งหมด เราจะไม่เพียงแค่เจาะจงที่ผลิตภัณฑ์ใดผลิตภัณฑ์หนึ่ง แต่จะเปรียบเทียบให้เห็นถึงจุดแข็งและจุดอ่อนของระบบที่หลากหลาย

แม้เทคโนโลยีด้านข้อมูลจะเปลี่ยนแปลงอย่างรวดเร็ว แต่หลักการพื้นฐานนั้นยังคงอยู่ หากเราเข้าใจหลักการเหล่านั้น ก็จะสามารถมองออกว่าเครื่องมือแต่ละชิ้นควรอยู่ตรงไหน ใช้งานมันอย่างไรให้เกิดประสิทธิภาพสูงสุด และจะหลีกเลี่ยงกับดักของมันได้อย่างไร หนังสือเล่มนี้จึงโฟกัสที่หลักการเหล่านั้นเป็นสำคัญ

หนังสือเล่มนี้ไม่ใช่คู่มือสอนใช้เครื่องมือใดเครื่องมือหนึ่ง แต่ก็ไม่ใช้ตำราที่เต็มไปด้วยทฤษฎีแบบแห้งแล้งเช่นกัน เราจะไปดูตัวอย่างระบบข้อมูลที่ประสบความสำเร็จมากมาย ซึ่งเป็นรากฐานของแอปพลิเคชันยอดนิยมที่เราใช้กันอยู่ และต้องแบกรับความต้องการด้าน **ความสามารถในการปรับขนาดได้ (Scalability)**, **ประสิทธิภาพ** และ **ความน่าเชื่อถือ (Reliability)** ในการใช้งานจริงทุกวัน เราจะเจาะลึกถึงกลไกภายใน, แกะรอยอัลกอริทึมสำคัญ และวิพากษ์**สิ่งที่ต้องแลกเปลี่ยน (Trade-off)** ที่พวกเขาตัดสินใจเลือก บนเส้นทางนี้ เราจะพยายามค้นหาแนวคิดที่เป็นประโยชน์เกี่ยวกับระบบข้อมูล ไม่ใช่แค่ดูว่ามันทำงานอย่างไร แต่ต้องรู้ว่า **ทำไมมันถึงทำงานแบบนั้น**

หลังจากอ่านจบ คุณจะมีความพร้อมในการตัดสินใจว่าเทคโนโลยีประเภทไหนเหมาะกับงานแบบใด และเข้าใจวิธีนำเครื่องมือต่างๆ มาประกอบรวมกันเป็นรากฐานสถาปัตยกรรมแอปพลิเคชันที่แข็งแกร่ง คุณจะพัฒนาสัญชาตญาณในการมองสิ่งที่ระบบทำภายในกล่องดำ เพื่อจะสามารถวิเคราะห์พฤติกรรม, ตัดสินใจออกแบบได้อย่างแม่นยำ และแกะรอยปัญหาที่อาจเกิดขึ้นได้อย่างมีอ้าชีพ

ปรัชญาหลักของหนังสือเล่มนี้คือ การรวบรวมมุมมองที่หลากหลาย ทั้งทฤษฎีและปฏิบัติ ทั้งเรื่องใหม่และเรื่องเก่า อุตสาหกรรมคอมพิวเตอร์มักจะหลงใหลในของใหม่ที่ดูหรูหรา และดูแคลนไอเดียที่ถือว่าเก่าหรือเพียงเพราะมันเป็นเพียงเอกสารงานวิจัย แต่นั่นคือความผิดพลาด เพราะไอเดียพื้นฐานที่ทรงพลังหลายอย่างในโลกคอมพิวเตอร์เกิดมาจากการงานวิจัย ทั้งที่เพิ่งเกิดขึ้นไม่นาน และที่ผ่านมาแล้วหลายทศวรรษ ในทางกลับกัน แวดวงวิชาการบางครั้งก็มักขาดความเข้าใจว่าประเด็นใดสำคัญในเชิงปฏิบัติและในโลกจริง หนังสือเล่มนี้จึงรวมสิ่งที่ดีที่สุดของทั้งสองโลกเข้าด้วยกัน ซึ่งได้แก่ ความแม่นยำและการเก็บรายละเอียดแบบวิชาการ ผสมผสานกับความใส่ใจในแง่การใช้งานจริงในระดับอุตสาหกรรม

ใครควรอ่านหนังสือเล่มนี้?

หากคุณมีคุณสมบัติตรงกับข้อใดข้อหนึ่งต่อไปนี้ หนังสือเล่มนี้จะมีค่าสำหรับคุณแน่นอน

- **วิศวกรซอฟต์แวร์, สถาปนิกซอฟต์แวร์ หรือผู้จัดการฝ่ายเทคนิค** ที่ต้องตัดสินใจเกี่ยวกับสถาปัตยกรรมของระบบที่ดูแล เช่น ต้องเลือกเครื่องมือมาแก้ปัญหา และหาวิธีนำไปใช้ให้ดีที่สุด โดยเฉพาะในระบบหลังบ้าน (Backend)

- **วิศวกรข้อมูล** ที่ต้องการเข้าใจบริบทที่กว้างขึ้นของระบบที่ต้องรับมือ หรือเป็น**วิศวกรคลาวด์ (Cloud Engineer)** ที่อยากเข้าถึงแก่นแท้ของระบบที่กำลังใช้งาน คุณจะพบว่าแม้ระบบแบบกระจาย (Distributed System) สมัยใหม่จะซ่อนความซับซ้อนไว้มากมาย แต่การเข้าใจหลักการพื้นฐานนั้นมีประโยชน์มหาศาลในการออกแบบและดีบั๊ก

- **ผู้ต้องการเรียนรู้ “วิธีทำให้ระบบข้อมูลปรับขนาดได้” (Scalability)** เช่น เพื่อรองรับแอปที่มีผู้ใช้หลายล้านคน, มีความพร้อมใช้งานสูง, ลดระยะเวลาที่ระบบหยุดทำงาน, ทำงานได้แข็งแกร่งทนทานต่อความล้มเหลว, บำรุงรักษาได้ง่ายในระยะยาว...แม้ว่าระบบจะเติบโตขึ้นหรือความต้องการและเทคโนโลยีจะเปลี่ยนไป

- **ผู้กำลังเตรียมตัวสัมภาษณ์งานด้าน “System Design”** ที่มักจะถูกถามให้ร่างสถาปัตยกรรมแอปพลิเคชัน และคุณจำเป็นต้องเรียนรู้หลักการของสถาปัตยกรรมข้อมูลที่ดี (ดังกระแสนวนโน้ตชีตที่โพสต์ว่า เขาได้งานในบริษัทไอทีชั้นนำ ก็เพราะหนังสือเล่มนี้)

- **ผู้อยากรู้ว่ามีอะไรเกิดขึ้นเบื้องหลัง** เว็บไซต์ใหญ่ๆ และบริการออนไลน์ยอดนิยม รวมถึงภายในฐานข้อมูลและระบบประมวลผลข้อมูลต่างๆ โดยเฉพาะหากคุณเป็นคนประเภทที่ชอบเจาะลึกมากกว่าแค่คำโฆษณา เพื่อให้ได้ความเข้าใจทางเทคนิคที่แม่นยำและลึกซึ้งถึงสิ่งที่ต้องแลกเปลี่ยนสำหรับแต่ละเทคโนโลยี

หนังสือเล่มนี้ตั้งสมมติฐานว่า คุณมีประสบการณ์ในการสร้างแอปพลิเคชันบนเว็บมาบ้างแล้ว และคุ้นเคยกับฐานข้อมูลเชิงสัมพันธ์และ SQL รวมถึงมีความเข้าใจพื้นฐานเกี่ยวกับโปรโตคอลเน็ตเวิร์คอย่าง TCP และ HTTP ส่วนคุณจะใช้ภาษาโปรแกรมหรือเฟรมเวิร์คใดนั้นไม่ใช่ประเด็นสำคัญสำหรับหนังสือเล่มนี้

มีอะไรใหม่ในฉบับปรับปรุงครั้งที่สอง?

ในฉบับปรับปรุงครั้งที่สองนี้ ยังคงมีเป้าหมายและขอบเขตเดิมเหมือนฉบับแรกที่ดีพิมพ์ในปี 2017 อย่างไรก็ตาม เราได้ปรับปรุงเนื้อหาอย่างละเอียดทั่วทั้งเล่ม เพื่อสะท้อนความเปลี่ยนแปลงทางเทคโนโลยีในทศวรรษที่ผ่านมา และเพิ่มความชัดเจนในการอธิบาย

ความเปลี่ยนแปลงทางเทคนิคที่ส่งผลต่อหนังสือเล่มนี้คือ การเติบโตอย่างก้าวกระโดดของ AI และการผุดขึ้นของสถาปัตยกรรมระบบข้อมูลแบบ **คลาวด์เนทีฟ (Cloud Native)** แม้หนังสือเล่มนี้จะไม่ใช้เรื่อง AI โดยตรง แต่เราได้เพิ่มเนื้อหาครอบคลุมระบบข้อมูลที่สนับสนุนงานด้าน AI และ **แมชชีนเลิร์นนิง (Machine Learning)** เข้าไปด้วย รวมถึง **เวกเตอร์เอ็มเบดดิ้ง (Vector Embedding)** (ที่ใช้สำหรับ Semantic Search), **ดาต้าเฟรม (DataFrame)** (สำหรับเตรียมชุดข้อมูลสำหรับเทรนโมเดล) และระบบการประมวลผลแบบแบดจ์ เพื่อจัดการข้อมูลเทรนปริมาณมหาศาล ส่วนแนวคิดแบบคลาวด์เนทีฟ เช่น การสร้างระบบข้อมูลบน **ออบเจกต์สโตร์ (Object Store)** แทนที่จะเป็นดิสก์ในเครื่อง ก็ได้สอดแทรกไว้ตลอดเล่มแล้ว

เรายังได้เพิ่มเรื่อง **Sync Engine** และ **Local-First Software**, **Workflow Engine** และ **Durable Execution**, **วิธีการเชิงรูปแบบ (Formal Method)** และ **การทดสอบแบบสุ่ม (Randomized Testing)**, **GraphQL** และเทคโนโลยีอื่นๆ ที่ควรค่าแก่การศึกษา นอกจากนี้ เรายังใส่บริบทด้านกฎหมายเข้าไปด้วย โดยการสำรวจผลกระทบของกฎหมายคุ้มครองข้อมูลส่วนบุคคล (GDPR/PDPA) และกฎหมายที่เกี่ยวข้อง ขณะเดียวกันเราได้ตัดบางอย่างออกไป เช่น MapReduce ที่ปัจจุบันแทบจะล้าสมัยแล้ว เราจึงเขียนบทที่เกี่ยวกับการประมวลผลแบบแบดจ์ใหม่ทั้งหมด และตัดแผนที่สโตร์ **Tolkien (The Lord of the Rings)** ออกไปอย่างน่าเสียดาย

โครงสร้างบางส่วนถูกจัดระเบียบใหม่และมีการเปลี่ยนลำดับบท บางบทมีการแก้ไขเพียงเล็กน้อย ในขณะที่บางบท (เช่น บทที่ 10 เรื่องความสอดคล้องของข้อมูลและฉันทามติ) ถูกเขียนใหม่แทบทั้งหมดเพื่อให้ชัดเจนยิ่งขึ้น

คำนำจากผู้เรียบเรียง

ทำไมวิศวกรระดับโลกถึงยกให้หนังสือเล่มนี้เป็น “คัมภีร์”

“หนังสือเล่มนี้ไม่ได้สอนใช้แค่ Database แต่สอนให้คิดเหมือน...คนออกแบบ Google, Facebook, Shopee, Amazon และ Netflix”

หากคุณหยิบหนังสือเล่มนี้ขึ้นมา เพื่อหวังเรียนรู้วิธีใช้ PostgreSQL, MySQL, MongoDB, Kafka หรือ Redis คุณอาจผิดหวัง

เพราะหนังสือเล่มนี้ไม่ได้สอนคำสั่ง SQL ไม่ได้สอนการติดตั้งคลัสเตอร์ และไม่ได้เอนเอียงไปทางเฟรมเวิร์คใดเป็นพิเศษ เพราะสิ่งที่ Martin Kleppmann และ Chris Riccomini ต้องการถ่ายทอดคือ “เหตุผลเบื้องหลัง” ของการออกแบบเทคโนโลยีเหล่านั้น

ตลอดเวลาเกือบ 40 ปีในวงการนี้ ผมพบว่าคนจำนวนมากรู้จัก “เครื่องมือ” แต่มีน้อยคนที่เข้าใจ “หลักการ” ที่อยู่เบื้องหลังเครื่องมือเหล่านั้น เรามักถกเถียงกันว่า PostgreSQL ดีกว่า MySQL หรือไม่, MongoDB เหมาะกว่าฐานข้อมูลเชิงสัมพันธ์หรือเปล่า, Kafka จะมาแทน Message Queue ได้จริงไหม หรือ Redis ควรใช้เป็น Cache หรือ Database

คำถามเหล่านี้ล้วนเป็นคำถามระดับ “ผลิตภัณฑ์” แต่หนังสือเล่มนี้กลับพาผู้อ่านย้อนกลับไปตั้งคำถามที่ลึกกว่านั้น

- เหตุใดระบบฐานข้อมูลจึงต้องมี Transaction?
- เหตุใด Distributed Transaction จึงมีต้นทุนสูง?
- เหตุใด Event Log จึงกลายเป็นหัวใจของระบบสมัยใหม่?
- เหตุใดองค์กรระดับโลก จึงค่อยๆ เปลี่ยนจาก Monolithic Database ไปสู่ Dataflow Architecture?
- และที่สำคัญที่สุด... เมื่อไรเราควรใช้แนวทางไหน?

นี่คือคุณค่าที่หาได้ยากจากหนังสือด้านฐานข้อมูลทั่วไป

หนังสือด้าน Database ส่วนใหญ่จะสอน “วิธีใช้” แต่หนังสือเล่มนี้สอน “วิธีคิด” ผู้เขียนแทบไม่เคยบอกว่าเทคโนโลยีใดดีที่สุด แต่เขากลับใช้พื้นที่หลายร้อยหน้าอธิบายว่า **ทุกการออกแบบมีข้อดี และทุกข้อดีก็แลกมาด้วยข้อเสีย**: B-tree แลกอะไรกับ LSM-tree, OLTP แลกอะไรกับ OLAP, Consistency แลกอะไรกับ Availability, Transaction แลกอะไรกับ Performance, Distributed System แลกอะไรกับ Complexity

เมื่ออ่านจบ คุณจะค้นพบว่า การออกแบบระบบไม่มีคำตอบสำเร็จรูป มีเพียง Trade-off ที่วิศวกรต้องเข้าใจ และตัดสินใจให้เหมาะกับบริบทของตนเอง

เหตุผลหนึ่งที่ทำให้ผู้คนวงการไอทีทั่วโลก (และผม) ยกให้หนังสือเล่มนี้เป็น “คัมภีร์” ไม่ใช่เพราะครอบคลุมฐานข้อมูลหลายชนิด แต่เพราะเชื่อมโยงศาสตร์หลายแขนงเข้าด้วยกัน อย่างน่าทึ่ง

หนังสือเล่มนี้เริ่มต้นจาก Database

แล้วค่อยๆ พาผู้อ่านเดินผ่าน Replication, Distributed Systems, Consensus, Batch Processing, Stream Processing, Event-Driven Architecture, CDC, Dataflow, Machine Learning, AI ไปจนถึงจริยธรรมของการใช้ข้อมูล

ระหว่างทาง คุณจะเริ่มมองเห็นว่า สิ่งที่เคยเรียนแยกกันเป็นวิชา ล้วนเชื่อมโยงถึงกัน

ฐานข้อมูลไม่ใช่เพียงที่เก็บข้อมูล, Event Log ไม่ใช่เพียง Message Queue, Cache ไม่ใช่เพียงตัวช่วยให้ระบบเร็วขึ้น, Search Index ไม่ใช่เพียงกลไกค้นหา, Machine Learning ก็ไม่ใช่เพียงแบบจำลองทางคณิตศาสตร์

แต่ทั้งหมดคือ “ข้อมูลแปรรูป (Derived Data)” ที่เกิดจากข้อมูลต้นทางเดียวกัน

สำหรับผม หนังสือเล่มนี้ได้เปลี่ยน **โลกทัศน์** ในการออกแบบของผมไปโดยสิ้นเชิง

.....

อีกสิ่งที่ทำให้หนังสือเล่มนี้แตกต่าง คือ ผู้เขียนไม่ได้หยุดอยู่ที่คำถามว่า “ระบบนี้ทำงานได้หรือไม่?” แต่ถามต่อว่า “ระบบนี้ควรถูกสร้างขึ้นหรือไม่?”

บทสุดท้ายของหนังสือ ไม่ได้พูดถึงประสิทธิภาพของฐานข้อมูล ไม่ได้พูดถึง Throughput หรือ Latency แต่กลับพูดถึง... อคติของอัลกอริทึม, ความเป็นส่วนตัว, การเฝ้าติดตามผู้คน, การเลือกปฏิบัติ, ความรับผิดชอบของวิศวกร และผลกระทบของซอฟต์แวร์ที่มีต่อสังคม

ผู้เขียนเตือนเราว่า ข้อมูลไม่ใช่เพียงสินทรัพย์ขององค์กร แต่คือเรื่องราวของผู้คน

หน้าที่ของวิศวกร ไม่ใช่เพียงสร้างระบบที่เร็วหรือรองรับผู้ใช้ได้มากขึ้น แต่ต้องสร้างระบบที่เคารพศักดิ์ศรีและความเป็นมนุษย์ด้วย ผมเชื่อว่านี่คือเหตุผลที่เล่มนี้ได้รับการยกย่องว่าเป็นมากกว่าหนังสือทางเทคนิค

ในฐานะผู้เรียบเรียง เล่มนี้ถือเป็นงานคิด-ค้น-เขียนที่ทำท้ายที่สุดเล่มหนึ่ง ไม่ง่ายเลยที่จะ **ถอดความคิด คิดหาคำไทย** (Author-Intent Semantic Transformation) ให้ได้ดีและง่ายที่สุด ความยากไม่ได้อยู่ที่ศัพท์เทคนิค หรือการตีความหลายชั้น เพราะค้นหาหรือสอบถามผ่านเสิร์ชผ่าน AI ก็ทำได้ แต่อยู่ที่ “วิธีคิดเขียน” ของ Martin และ Chris ซึ่งมีลักษณะเชิง

นามธรรมซับซ้อน เชื่อมโยงหลายแนวคิดหลายศาสตร์เข้าด้วยกันไปทั่วเล่ม บางประโยคก็ยาว หลายบรรทัด และแต่ละประโยคอาจมีทั้งคณิตศาสตร์ ทฤษฎี และปรัชญาการออกแบบ ที่ซ่อนไว้ระหว่างบรรทัดให้ตีความไปพร้อมกัน

ระหว่างการเรียบเรียง บางครั้งต้องย้อนอ่านซ้ำ สลับหยุดคิดหลายรอบ เพื่อตกผลึกทำความเข้าใจให้ได้ว่า **ผู้เขียนต้องการสื่ออะไร** แล้วจึงเรียบเรียงเขียนใหม่ โดยพยายามรักษาเจตนาของผู้เขียนและความหมายดั้งเดิม ไว้ให้มากที่สุดเท่าที่จะเป็นไปได้

.....

หากคุณเป็น Software Architect, Backend Engineer, Data Engineer, Platform Engineer, Cloud Architect หรือผู้ที่ต้องออกแบบระบบสารสนเทศขนาดใหญ่ ผมเชื่อว่าเล่มนี้จะไม่ใช่วินิจฉัยที่อ่านเพียงครั้งเดียว แล้ววางไว้บนชั้น

แต่จะเป็นเล่มที่คุณหยิบกลับมาเปิดซ้ำทุกครั้ง เมื่อเผชิญโจทย์ใหม่ในการออกแบบระบบ เพราะเทคโนโลยีอาจเปลี่ยนทุกปี

Framework อาจเปลี่ยนทุก 3 เดือน

ฐานข้อมูลรุ่นใหม่อาจเกิดขึ้นทุกวัน

แต่หลักการที่หนังสือเล่มนี้อธิบาย ยังคงใช้ได้เสมอ

และนี่คือเหตุผลที่ทั่วโลกยกย่องให้ *Designing Data-Intensive Applications* เป็นหนังสืออ้างอิงสำคัญที่สุดเล่มหนึ่งสำหรับ Software Architect, Data Engineer, Platform Engineer และ Backend Engineer และยังทรงคุณค่าแม้เทคโนโลยีจะเปลี่ยนไป เพราะสิ่งที่หนังสือถ่ายทอดคือ **หลักการที่อยู่เหนือเครื่องมือ** ไม่ใช่เพียงวิธีใช้เทคโนโลยีรุ่นใดรุ่นหนึ่งเท่านั้น (อยากให้ลองฟังบทสัมภาษณ์ของ Martin ที่ https://youtu.be/SVOrURyOu_U)

ผมเชื่อว่า **หนังสือเล่มนี้** ไม่ใช่เป็นเพียงแค่หนังสือด้านฐานข้อมูล แต่เป็นหนึ่งใน **หนังสือด้านสถาปัตยกรรมซอฟต์แวร์** ที่ล้ำหน้าเล่มหนึ่งของยุคนี้ สมควรมีไว้เป็นหนังสืออ้างอิงประจำโต๊ะทำงานของทุกคนที่ต้องการสร้างระบบซึ่ง **เชื่อถือได้ ขยายได้ และพร้อมรองรับการเปลี่ยนแปลงในอนาคต**

วิโรจน์ อัครรังสี

สารบัญ

บทที่ 1 สิ่งที่ต้องแลกเปลี่ยนในสถาปัตยกรรมระบบข้อมูล	21
ระบบเชิงปฏิบัติการ กับระบบเชิงวิเคราะห์	24
ลักษณะการประมวลผลเชิงธุรกรรมกับเชิงวิเคราะห์	26
ดาต้าแวร์เฮาส์	28
ระบบบันทึกหลักและข้อมูลแปรรูป	33
ระบบคลาวด์เทียบกับระบบที่ติดตั้งเอง	35
ข้อดีและข้อเสียของบริการคลาวด์	36
สถาปัตยกรรมระบบแบบ Cloud Native	37
งานดูแลระบบในยุคคลาวด์	40
ระบบแบบกระจายกับระบบแบบโหนดเดียว	42
ปัญหาของระบบกระจาย	43
ไมโครเซอร์วิสและเซิร์ฟเวอร์เลส	44
ระบบคลาวด์กับซูเปอร์คอมพิวเตอร์	47
ระบบข้อมูล, กฎหมาย และสังคม	48
สรุป	50
 บทที่ 2 ข้อกำหนดด้านคุณภาพของระบบ	 53
กรณีศึกษา: ไลน์หน้าแรกของโซเชียลเน็ตเวิร์ค	54
การแทนข้อมูล: ผู้ใช้ โพสต์ และการติดตาม	55
การสร้างและอัปเดตไทม์ไลน์จากข้อมูลต้นทาง	56
การวัดประสิทธิภาพของระบบ	58
เลเทนซีและเวลาตอบสนอง	61
ค่าเฉลี่ย, ค่ามัธยฐาน และเปอร์เซ็นต์ไทล์	63
การใช้ตัวชี้วัดเวลาในการตอบสนอง	66
ความเชื่อถือได้และการทนต่อความผิดพลาด	68
การทนต่อความผิดพลาด	69
ความผิดพลาดทางฮาร์ดแวร์และซอฟต์แวร์	70
มนุษย์กับความเชื่อถือได้ของระบบ	73

ความสามารถในการปรับขนาดระบบ	76
การเข้าใจภาระงาน	77
สถาปัตยกรรมแบบหน่วยความจำร่วม, ดิสก์ร่วม และไม่ใช่ทรัพยากรร่วม	78
หลักการปรับขนาดระบบ	79
ความสามารถในการบำรุงรักษา	80
การออกแบบให้ดูแลง่าย	81
การออกแบบที่เน้นความเรียบง่าย	82
การออกแบบให้พัฒนาปรับเปลี่ยนได้ง่าย	83
สรุป	84
บทที่ 3 โมเดลข้อมูลและภาษาคิวรี	87
โมเดลเชิงสัมพันธ์ กับโมเดลเอกสาร	89
ความไม่สอดคล้องระหว่างออบเจกต์และตาราง	90
การจัดรูปบรรทัดฐาน, การลดรูปบรรทัดฐาน และการเชื่อมตาราง	94
ความสัมพันธ์แบบหลายต่อหนึ่ง และหลายต่อหลาย	99
สตาร์และสโนว์เฟลก: สคีมาเพื่องานวิเคราะห์	102
เมื่อไรควรเลือกใช้โมเดลใด	104
โมเดลข้อมูลแบบกราฟ	109
กราฟคุณสมบัติ	112
ภาษา Cypher	114
การคิวรีข้อมูลกราฟด้วย SQL	116
โมเดลแบบ Triple Store และภาษา SPARQL	119
Datalog คิวรีเชิงสัมพันธ์แบบเรียกซ้ำ	124
GraphQL	127
การจัดเก็บเหตุการณ์และ CQRS	130
ดาต้าเฟรม, เมทริกซ์ และอาร์เรย์	135
สรุป	137
บทที่ 4 การจัดเก็บและเรียกใช้ข้อมูล	141
ระบบจัดเก็บและดัชนีสำหรับ OLTP	142
ระบบจัดเก็บ Log แบบมีโครงสร้าง	144

โครงสร้างข้อมูลแบบ B-Tree	153
เปรียบเทียบ B-Tree และ LSM-Tree	157
ดัชนีแบบหลายคอลัมน์และดัชนีรอง	161
การเก็บค่าไว้ในดัชนี	162
การเก็บทุกอย่างไว้ในหน่วยความจำ	163
การจัดเก็บข้อมูลเพื่องานวิเคราะห์	164
ดาต้าแวร์เฮาส์บนคลาวด์	165
การจัดเก็บข้อมูลแบบคอลัมน์	167
การคอมไพล์และประมวลผลแบบเวกเตอร์	174
วิธีที่จัดทำไว้ล่วงหน้า และดาต้าคิวบ์	176
ดัชนีแบบหลายมิติ และดัชนีข้อความแบบเต็ม	178
การค้นหาข้อความแบบเต็ม	180
เวกเตอร์เอ็มเบดดิ้ง	182
สรุป	185
 บทที่ 5 การเข้ารหัสและการพัฒนาปรับเปลี่ยนได้	189
ฟอร์แมตสำหรับการเข้ารหัสข้อมูล	192
ฟอร์แมตเฉพาะของแต่ละภาษา	193
JSON, XML และรูปแบบไบนารีที่เกี่ยวข้อง	194
Protocol Buffers	200
Avro	204
ข้อดีของสคีมา	211
รูปแบบของดาต้าโฟล์ว	212
ดาต้าโฟล์วผ่านฐานข้อมูล	213
ดาต้าโฟล์วผ่านเซอร์วิส REST และ RPC	215
การประมวลผลแบบคงทนและเวิร์คโฟล์ว	225
สถาปัตยกรรมที่ขับเคลื่อนด้วยเหตุการณ์	228
สรุป	232
 บทที่ 6 การทำสำเนาข้อมูล	235
การทำสำเนาแบบผู้นำเดี่ยว (Single-Leader Replication)	237

การทำสำเนาแบบซิงโครนัสและอะซิงโครนัส	239
การเพิ่มโหนดผู้ติดตามใหม่	241
การรับมือเมื่อโหนดล้มเหลว	244
วิธีทำ Replication Log	249
ปัญหาความล่าช้าในการทำสำเนา	252
แนวทางแก้ไขปัญหา Replication Lag	260
การทำสำเนาแบบหลายผู้นำ (Multi-Leader Replication)	261
การทำงานข้ามภูมิภาค	262
กลไกซิงก์ข้อมูลและซอฟต์แวร์ที่เน้นเครื่องของผู้ใช้	268
การจัดการกับการเขียนที่ขัดแย้งกัน	271
การทำสำเนาแบบไม่มีผู้นำ (Leaderless Replication)	280
การเขียนข้อมูลขณะโหนดบางตัวไม่พร้อมใช้งาน	281
เปรียบเทียบประสิทธิภาพการทำสำเนาแบบผู้นำเดียวกับแบบไม่มีผู้นำ	288
การทำงานข้ามหลายภูมิภาค	290
การตรวจจับการเขียนพร้อมกัน	291
สรุป	298
บทที่ 7 การแบ่งส่วนข้อมูล	303
ข้อดีและข้อเสียของการแบ่งส่วนข้อมูล	306
การแบ่งส่วนข้อมูลสำหรับระบบหลายผู้เช่า	307
การแบ่งส่วนข้อมูลรูปแบบคีย์-ค่า (Key-Value)	309
การแบ่งตามช่วงคีย์	310
การแบ่งตามค่าแฮชของคีย์	312
ปัญหางานไม่สมดุล และการลดฮอตสเปด	319
การปรับสมดุลแบบอัตโนมัติ และแบบจัดการเอง	320
การกำหนดเส้นทางของคำขอ	322
การแบ่งส่วนข้อมูล และดัชนีรอง	325
ดัชนีรองแบบเฉพาะส่วน	326
ดัชนีรองแบบภาพรวม	327
สรุป	330

บทที่ 8 รายการธุรกรรม	333
ทรานแซกชันคืออะไร	335
ความหมายของ ACID	336
ธุรกรรมแบบเดี่ยวและแบบหลายออบเจกต์	342
ระดับการแยกส่วนแบบฟอนปรน	348
การอ่านข้อมูลที่ยืนยันแล้ว	350
การแยกส่วนด้วยสแนปชอต และการอ่านซ้ำ	354
ป้องกันข้อมูลหายจากการอัปเดตซ้อน	362
ปัญหาจากการเขียนพร้อมกัน และปรากฏการณ์แฟนธอม	366
การเรียงลำดับได้แม้ทำงานพร้อมกัน	373
การประมวลผลแบบเรียงลำดับจริง	374
การล็อกแบบสองช่วง	379
การแยกธุรกรรมแบบสแนปชอตที่เรียงลำดับได้	384
ธุรกรรมแบบกระจาย	392
การยืนยันแบบสองช่วง	394
ธุรกรรมแบบกระจายข้ามระบบ	400
ธุรกรรมแบบกระจายภายในฐานข้อมูล	405
บทวนการประมวลผลข้อความให้มีผลเพียงครั้งเดียว	406
สรุป	407
 บทที่ 9 ปัญหาของระบบแบบกระจาย	 413
ความผิดพลาดและล้มเหลวบางส่วน	414
เครือข่ายที่ไม่น่าเชื่อถือ	416
ข้อจำกัดของ TCP	417
ปัญหาเครือข่ายในโลกจริง	419
การตรวจจับความผิดพลาด	420
การหมดเวลา และความล่าช้าที่ไม่มีขอบเขต	422
เครือข่ายแบบซิงโครนัส เทียบกับอะซิงโครนัส	425
นาฬิกาที่ไม่น่าเชื่อถือ	429
นาฬิกาโมโนโทนิก กับแบบตามเวลาจริง	430
การซิงโครไนซ์นาฬิกาและความแม่นยำ	432

การพึ่งพานาฬิกาที่ซิงโครไนซ์	434
การหยุดชั่วคราวของโปรเซส	440
ความรู้, ความจริง และความเท็จ	446
กฎเสียงข้างมาก	447
Lock และ Lease ในระบบแบบกระจาย	448
ฟอลต์แบบไบแซนไทน์	454
แบบจำลองพฤติกรรมระบบกับความเป็นจริง	458
วิธีตรวจสอบระบบด้วยหลักคณิตศาสตร์ และการทดสอบแบบสุ่ม	464
สรุป	470
บทที่ 10 ความสอดคล้องของข้อมูลและฉันทามติ	473
ความสอดคล้องตามลำดับเวลาจริง (Linearizability)	475
อะไรทำให้ระบบมีลิเนียร์ซอร์อะบิลิตี?	477
การพึ่งพา Linearizability	483
การพัฒนาาระบบที่มีคุณสมบัติเป็น Linearizable	487
ต้นทุนของ Linearizability	490
ตัวสร้าง ID และนาฬิกาเชิงตรรกะ	495
นาฬิกาเชิงตรรกะ	498
ตัวสร้าง ID แบบ Linearizable	502
ฉันทามติ	505
ฉันทามติในหลายรูปแบบ	507
ฉันทามติในทางปฏิบัติ	516
บริการประสานงาน	524
สรุป	528
บทที่ 11 การประมวลผลแบบแบตช์	533
การประมวลผลแบบแบตช์ด้วยเครื่องมือของ Unix	537
การวิเคราะห์ Log แบบง่าย	538
การเชื่อมต่อคำสั่ง เทียบกับการเขียนโปรแกรมเอง	539
การเรียงลำดับ เทียบกับการรวมข้อมูลในหน่วยความจำ	540
การประมวลผลแบบแบตช์ในระบบแบบกระจาย	541

ระบบไฟล์แบบกระจาย	542
ระบบจัดเก็บข้อมูลแบบออบเจกต์	544
การประสานงานการประมวลผลแบบกระจาย	546
โมเดลการประมวลผลแบบแบตช์	552
MapReduce	553
ดาต้าโพล์เฮนจิน	555
การสับเปลี่ยนข้อมูล	556
การเชื่อมและจัดกลุ่มข้อมูล	558
ภาษาสำหรับการคิวรี	560
ดาต้าเฟรม	563
กรณีการใช้งานแบตช์	564
การสกัด, แปลง และโหลดข้อมูล (ETL)	564
การวิเคราะห์ข้อมูล	565
แมชชีนเลิร์นนิง	566
การให้บริการข้อมูลแปรรูป	568
สรุป	570
บทที่ 12 การประมวลผลสตรีม	573
การส่งสตรีมเหตุการณ์	574
ระบบรับส่งข้อความ	576
ตัวกลางรับส่งข้อความแบบอิงล็อก	583
ฐานข้อมูลและสตรีม	590
การทำให้ระบบสอดคล้องกัน	591
การตรวจจับการเปลี่ยนแปลงข้อมูล	593
สถานะ, สตรีม และข้อมูลที่ไม่เปลี่ยนแปลง	600
การประมวลผลสตรีม	607
การประยุกต์ใช้การประมวลผลสตรีม	608
การจัดการเรื่องเวลา	614
การเชื่อมสตรีม	619
ความทนทานต่อความผิดพลาด	623
สรุป	627

บทที่ 13 ปรัชญาของระบบสตรีม	631
การบูรณาการข้อมูล	632
การเชื่อมต่อเครื่องมือเฉพาะทางด้วยข้อมูลแปรรูป	632
การประมวลผลแบบแบตช์และสตรีม	637
การแยกองค์ประกอบของฐานข้อมูล	640
การประกอบเทคโนโลยีจัดเก็บข้อมูลเข้าด้วยกัน	641
การออกแบบแอปพลิเคชันที่อิงกับดาต้าโพล์	645
การสังเกตสถานะของข้อมูลแปรรูป	650
เป้าหมายคือความถูกต้อง	656
แนวคิดของความถูกต้องตั้งแต่ต้นทางถึงปลายทาง	658
การบังคับใช้ข้อกำหนด	663
ความทันเวลาและถูกต้องครบถ้วน	668
เชื่อใจ แต่ต้องตรวจสอบ	672
สรุป	676
 บทที่ 14 การใช้เทคโนโลยีอย่างรับผิดชอบ	679
การวิเคราะห์เชิงพยากรณ์	680
อคติและการเลือกปฏิบัติ	681
หน้าที่และความรับผิดชอบต่อผลลัพธ์	682
วงจรป้อนกลับ	683
ความเป็นส่วนตัวและการติดตามผล	684
การเฝ้าติดตาม	685
ความยินยอมและเสรีภาพในการเลือก	687
ความเป็นส่วนตัวและการใช้ข้อมูล	688
ข้อมูลในฐานะของสินทรัพย์และอำนาจ	690
บทเรียนจากการปฏิบัติอุตสาหกรรม	691
กฎหมายและการกำกับดูแลตนเอง	692
สรุป	694