# Bài tập Chương 12

## Bài tập 12.1

* Tìm kiếm Web (Web Search): Chủ yếu dựa trên từ khóa. Mục tiêu là tìm các tài liệu/trang web có liên quan và xếp hạng chúng. Ngôn ngữ truy vấn đơn giản và kết quả là một danh sách các liên kết (URL).
* Truy vấn Web (Web Querying): Sử dụng một ngôn ngữ khai báo, có cấu trúc (như SQL, SPARQL). Mục tiêu là trích xuất dữ liệu hoặc sự kiện cụ thể từ web bằng cách xem web như một cơ sở dữ liệu (thường là bán cấu trúc). Kết quả là dữ liệu có cấu trúc, không chỉ là liên kết.

## Bài tập 12.2

Một kiến trúc cho máy tìm kiếm trên cluster shared-nothing:

* Crawler: Chạy song song trên nhiều node. Một node điều phối trung tâm quản lý hàng đợi URL và phân công nhiệm vụ cho các crawler.
* Page Repository: Lưu trữ các trang đã thu thập trên một hệ thống file phân tán như HDFS. HDFS sẽ tự động quản lý việc phân mảnh (thành các chunk) và sao chép.
* Indexer: Thực thi như một job MapReduce/Spark. Các map task phân tích trang và phát ra các cặp (term, docID). Các reduce task thu thập các cặp này để xây dựng các phân vùng của chỉ mục đảo ngược (inverted index).
* Query Engine: Một tập hợp các node đầu cuối tiếp nhận truy vấn của người dùng và điều phối việc thực thi.
* Phân mảnh (Partitioning): Chỉ mục đảo ngược được phân mảnh theo term (term-based partitioning). Tức là, tất cả các thông tin cho một từ khóa (ví dụ: “database”) sẽ được lưu trữ trên cùng một node (hoặc một tập node).
* Sao chép (Replication): Mỗi phân vùng của chỉ mục và dữ liệu trong Page Repository được sao chép trên nhiều node để đảm bảo khả năng chịu lỗi và cân bằng tải cho các truy vấn đọc.

## Bài tập 12.3

Chiến lược thực thi song song cho một truy vấn từ khóa Q:

1. Truy vấn Q đến một node Query Engine (đóng vai trò điều phối).
2. Node điều phối gửi Q đến tất cả các node đang giữ một phân vùng của chỉ mục đảo ngược.
3. Mỗi node tìm kiếm cục bộ trên phân vùng chỉ mục của mình, lấy ra danh sách các docID tương ứng với các từ khóa và thực hiện các phép toán (giao/hợp) để có được tập docID kết quả cục bộ.
4. Mỗi node gửi tập docID cục bộ của mình về cho node điều phối.
5. Node điều phối hợp nhất tất cả các tập docID lại.
6. Node điều phối áp dụng thuật toán xếp hạng (ví dụ: sử dụng giá trị PageRank đã được tính toán trước) cho danh sách docID cuối cùng, lấy thông tin tóm tắt (snippet) từ Page Repository cho các kết quả top-k, và trả về cho người dùng.

## Bài tập 12.4

Kiến trúc mở rộng cho nhiều site địa lý:

* Sao chép: Toàn bộ Page Repository và toàn bộ chỉ mục đảo ngược (đã được phân mảnh) được sao chép đầy đủ (fully replicated) tại mỗi site (mỗi trung tâm dữ liệu).
* Đồng bộ dữ liệu: Sử dụng một giao thức sao chép lười biếng (lazy replication). Việc thu thập và đánh chỉ mục có thể diễn ra ở một site chính, sau đó các bản cập nhật cho chỉ mục và các trang mới sẽ được định kỳ lan truyền đến các site khác.
* Định tuyến truy vấn: Truy vấn của người dùng được định tuyến đến trung tâm dữ liệu gần nhất về mặt địa lý (ví dụ: sử dụng định tuyến dựa trên DNS) để giảm thiểu độ trễ.

## Bài tập 12.5

Chiến lược thực thi song song cho một truy vấn tìm kiếm từ khóa trên kiến trúc đa site giống hệt với chiến lược trong bài 12.3. Tuy nhiên, toàn bộ quá trình này được thực thi hoàn toàn bên trong một trung tâm dữ liệu duy nhất – trung tâm dữ liệu gần người dùng nhất mà truy vấn đã được định tuyến đến.

## Bài tập 12.6

* Hạn chế của tích hợp quan hệ:
  + Không khớp cấu trúc: Ánh xạ 1-N giữa Name trong EMP1 và {Firstname, Lastname} trong EMP2 khó biểu diễn trong một view quan hệ phẳng mà không dùng các hàm xử lý chuỗi.
  + Mất dữ liệu: Nếu một nhân viên chỉ tồn tại ở một nguồn, phép nối (JOIN) có thể làm mất thông tin của nhân viên đó.
* Lược đồ XML tích hợp: XML xử lý sự không đồng nhất về cấu trúc tốt hơn.

<xs:element name="EMP">  
 <xs:complexType>  
 <xs:sequence>  
 <xs:element name="Name">  
 <xs:complexType>  
 <xs:choice>  
 <xs:element name="FullName" type="xs:string"/>  
 <xs:sequence>  
 <xs:element name="FirstName" type="xs:string"/>  
 <xs:element name="LastName" type="xs:string"/>  
 </xs:sequence>  
 </xs:choice>  
 </xs:complexType>  
 </xs:element>  
 <xs:element name="City" type="xs:string"/>  
 <xs:element name="Phone" type="xs:string" minOccurs="0"/>  
 </xs:sequence>  
 </xs:complexType>  
</xs:element>

Lược đồ này cho phép tên nhân viên được biểu diễn dưới dạng một chuỗi FullName (từ EMP1) hoặc các phần tử FirstName và LastName riêng biệt (từ EMP2), giải quyết được sự không đồng nhất.

## Bài tập 12.7

| Loại xung đột | Mô tả |
| --- | --- |
| Xung đột kiểu (Type Conflicts) | Cùng một đối tượng được biểu diễn bằng một thuộc tính trong lược đồ này nhưng lại là một thực thể trong lược đồ khác. |
| Xung đột phụ thuộc (Dependency Conflicts) | Cùng một mối quan hệ được biểu diễn bằng các bản số khác nhau (ví dụ 1-N so với M-N). |
| Xung đột khóa (Key Conflicts) | Các khóa chính khác nhau được chọn cho cùng một thực thể trong các lược đồ khác nhau. |
| Xung đột hành vi (Behavioral Conflicts) | Các thao tác CSDL (ví dụ: xóa) gây ra các hiệu ứng phụ khác nhau trong các hệ thống khác nhau. |

## Bài tập 12.8

Trong mô hình mạng xã hội:

* Authority (Trang uy tín): Hồ sơ của một người nổi tiếng hoặc một chuyên gia đầu ngành. Họ có nhiều người theo dõi (hub) trỏ đến.
* Hub (Trang trung tâm): Trang tổng hợp tin tức hoặc một blog uy tín thường xuyên trỏ đến các nguồn tin/chuyên gia chất lượng (authority).

## Bài tập 12.9

1. bib.doc.(authors|author)
2. bib.doc.#.title
3. bib.doc.chapters.chapter+

## Bài tập 12.10

1. SPARQL:

PREFIX movie: <http://data.linkedmdb.org/resource/movie/>  
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>  
  
SELECT ?title  
WHERE {  
 ?film movie:director <http://data.linkedmdb.org/resource/director/8476> .  
 ?film rdfs:label ?title .  
}

1. XQuery (giả sử có cấu trúc XML tương tự):

for $film in doc("mdb.xml")//film  
where $film/director/@id = "8476"  
return $film/title/text()

## Bài tập 12.11

Không, không phải tất cả các hệ thống con đều cần thiết.

* Hệ thống đơn giản (ví dụ: file sharing): Chỉ cần lớp P2P Network Sublayer để định tuyến và truyền file.
* Hệ thống có truy vấn (ví dụ: DHT): Cần Query Manager và P2P Network Sublayer.
* Hệ thống tích hợp dữ liệu không đồng nhất: Cần đầy đủ các thành phần, bao gồm cả Semantic Mappings và Wrapper.
* Hệ thống chỉ đọc: Không cần Update Manager.
* Hệ thống không cache: Không cần Cache Manager.

## Bài tập 12.12

get(4378) Kết quả sẽ là danh sách các con trỏ đến các peer chứa bản sao của đối tượng, ví dụ: {4228, AA93}.

## Bài tập 12.13

Trong Tapestry, một đối tượng được sao chép bằng cách insert cùng một object\_id tại nhiều server node khác nhau. Mỗi thao tác insert này sẽ tạo ra một cây định tuyến riêng biệt từ server node đến root node của object\_id. Khi thực hiện lookup, truy vấn sẽ được định tuyến về phía root node và sẽ dừng lại ở con trỏ đầu tiên mà nó gặp trên đường đi. Do đó, truy vấn sẽ được tự động định tuyến đến bản sao “gần nhất” về mặt logic trên topo của Tapestry.

## Bài tập 12.14

Blockchain công khai (public) và riêng tư (private) khác nhau cơ bản về cơ chế đồng thuận.

* Public Blockchain (ví dụ: Bitcoin):
  + Vấn đề: Các thành viên không được biết và không tin tưởng lẫn nhau, có thể có các hành vi phá hoại (Byzantine failures).
  + Giao thức xác thực: Cần một giao thức chống lại tấn công Byzantine, thường là Proof-of-Work (PoW). Giao thức này rất tốn kém về mặt tính toán để đảm bảo an toàn trong môi trường không tin cậy.
* Private Blockchain (Permissioned):
  + Vấn đề: Các thành viên đã được xác thực và có một mức độ tin tưởng nhất định. Lỗi thường được giả định là lỗi crash (crash failures), không phải lỗi Byzantine.
  + Giao thức xác thực: Có thể sử dụng các giao thức đồng thuận hiệu quả hơn nhiều như Paxos hoặc Raft, vốn được thiết kế cho môi trường có kiểm soát và chịu được lỗi crash.

## Bài tập 12.15

Có. Nếu các trang HTML được đánh dấu (markup) bằng RDFa (RDF in attributes) hoặc microdata, một crawler có thể trích xuất các bộ ba (triples) RDF trực tiếp từ các trang này. Sau khi thu thập, các bộ ba này có thể được tải vào một kho dữ liệu RDF (RDF store). Sau đó, người dùng có thể thực hiện các truy vấn SPARQL phức tạp trên kho dữ liệu này để khai thác các mối quan hệ ngữ nghĩa đã được trích xuất, vượt xa khả năng của tìm kiếm từ khóa.

## Bài tập 12.16

|  | Web Tables/Fusion Tables | Linked Open Data (LOD) |
| --- | --- | --- |
| Mô hình dữ liệu | Quan hệ (bảng 2 chiều). | Đồ thị (Graph - các bộ ba RDF). |
| Lược đồ | Lược đồ được suy ra từ cấu trúc bảng. | Dựa trên các ontology và từ vựng (ví dụ RDFS, OWL). |
| Liên kết dữ liệu | Dựa trên việc nối các bảng có các cột chung. | Dựa trên các liên kết RDF (URI) giữa các thực thể trong các bộ dữ liệu khác nhau. |
| Phạm vi | Tích hợp các bảng dữ liệu có cấu trúc. | Tích hợp dữ liệu web ở quy mô toàn cầu, tạo ra một “Web of Data”. |

## Bài tập 12.17

1. Duyệt và Thu thập (Crawling & Harvesting): Sử dụng một crawler chuyên dụng để duyệt các nguồn OAI-PMH, gửi các yêu cầu ListRecords để thu thập siêu dữ liệu (thường ở định dạng Dublin Core XML).
2. Chuyển đổi (Transformation): Viết các kịch bản (ví dụ: dùng XSLT) để chuyển đổi siêu dữ liệu XML thu thập được sang định dạng RDF. Mỗi bản ghi siêu dữ liệu sẽ được chuyển thành một tập các bộ ba RDF.
3. Lưu trữ (Storage): Tải các bộ ba RDF đã tạo vào một kho dữ liệu RDF (RDF triple store).
4. Truy vấn (Querying): Cung cấp một SPARQL endpoint cho phép người dùng thực hiện các truy vấn có cấu trúc trên bộ dữ liệu RDF đã được tích hợp.

## Bài tập 12.18

db.collection.find( { "comments.who": "jane" } )

## Bài tập 12.19

db.collection.find( { "tags": { $all: ["business", "ramblings"] } } ) Toán tử $all đảm bảo rằng mảng tags chứa tất cả các phần tử được chỉ định.