Mô hình dữ liệu cho ứng dụng blog đơn giản trong cơ sở dữ liệu quan hệ

* Sẽ là sai lầm lớn nếu chúng ta tiếp cận với document database khi cố gắng đi theo mô hình quan hệ và áp dụng nó vào document database. Đoạn code dưới đây là dữ liệu chúng ta muốn lưu trữ:

|  |
| --- |
| var user = new User("ayende");  var blog = new Blog("Ayende @ Rahien", user) { Tags = {".NET", "Architecture", "Databases" } };  var categoryRaven = new Category("Raven");  var categoryNoSQL = new Category("NoSQL");  var post = new Post(blog, "RavenDB", "... content ...")  {  Categories = { categoryRaven, categoryNoSQL },  Tags = {"RavenDB", "Announcements" }  };  var comment = new Comment(post, "Great news");  PersistAll(user, blog, categoryRaven, categoryNoSQL, post, comment); |

* Chúng ta sẽ sử dụng code để biểu diễn dữ liệu mà không ràng buộc dữ liệu vào định dạng cụ thể nào.
* Lưu dữ liệu một cách sai lầm là lưu trữ dữ liệu mỗi đối tượng như một document riêng biệt, tương tự như cách chúng ta lưu trữ từng đối tượng thành một hàng riêng trong cơ sở dữ liệu quan hệ. Lưu trữ không đúng cách trông sẽ như bên dưới:

|  |  |  |
| --- | --- | --- |
| |  | | --- | | // users/ayende  {  "type": "user",  "name": "ayende"  }  // tags/1  {  "name": ".NET"  }  // tags/2  {  "name": "Architecture"  }  // tags/3  {  "name": "Databases"  }  // tags/4  {  "name": "RavenDB"  }  // tags/5  {  "name": "Announcements"  } | | // categories/1  {  "name": "Raven"  }  // categories/2  {  "name" : "NoSQL"  }  // blogs/1  {  "type": "blog",  "users": ["users/ayende"],  "name": "Ayende @ Rahien",  "tags": ["tags/1", "tags/2", "tags/3"]  }  // posts/1  {  "blog": "blogs/1",  "title": "RavenDB",  "content": "... content ...",  "categories": ["categories/1", "categories/2"]  "tags" : ["tags/4", "tags/5"]  }  // comments/1  {  "post": "posts/1",  "content": "Great News"  } | |

* Lưu ý: không nên mô hình một document database theo cách này.
* Lý do chính là: điều này sẽ sai vì một document database không thật sự hỗ trợ việc joins, unions hay bất cứ thứ gì làm cho mô hình làm việc hiệu quả trong mô hình cơ sở dữ liệu.
* Một số yêu cầu khi hiển thị dữ liệu trong ứng dụng:
  + Main page: show list of blogs
  + Main page: show list of recent posts
  + Main page: show list of recent comments
  + Main page: show tag cloud for posts
  + Main page: show categories
  + Post page: show post and all comments
  + Post page: add comment to post
  + Tag page: show all posts for tag
  + Categories page: show all posts for category
* Main page: show list of blogs
  + Sử dụng SQL:

|  |
| --- |
| select \* from blogs |

* + Sử dụng DocDB, sử dụng index có sẵn để truy vấn dữ liệu dựa vào kiểu dữ liệu:

|  |
| --- |
| docDb.Query<Blog>("DocumentsByType", query:"type:blog"); |

* Main page: show list of recent posts
  + Sử dụng SQL:

|  |
| --- |
| select \* from posts order by PostedAt desc |

* + Sử dụng DocDB, sử dụng index do chúng ta định nghĩa, cho phép sắp xếp trong nó(không khó khăn lắm):
    - Index fuction:

|  |
| --- |
| from doc in docs  where doc.type == "post"  select new {doc.posted\_at} |

* + - Truy vấn dữ liệu sử dụng:

|  |
| --- |
| docDb.Query<Post>("Posts", orderBy:"-posted\_at"); |

* Main page: show list of recent comments: tương tự recent posts
* Main page: show tag cloud for posts
  + Sử dụng SQL:

|  |
| --- |
| select Name, COUNT(\*) as TagCount from tags  where ItemType = 'Posts'  group by Name |

* + Sử dụng DocDB, chúng ta cần viết một map/reduce index:
    - Map/ reduce index:

|  |
| --- |
| from doc in docs  where doc.type == "posts"  from tag in doc.tags  select new { tag, count = 1 }  from result in results  group result by result.tag into g  select new { tag = g.Key, count = g.Sum(x=>x.count) } |

* + - Khi đã có map/reduce index chúng ta sử dụng nó để lấy dữ liệu:

|  |
| --- |
| var tagCloud = new TagCloud();  var tagIds = docDb.Query<TagAndCount>("TagsCloud", orderBy:"+count");  foreach(var tagId in tagIds)  {  var tag = docDb.Get<Tag>(tagId.Tag);  tagCloud.Add(tag.Name, tagId.Count);  } |

* Main page: show categories: tương tự blogs
* Post page: show post and all comments
  + Sử dụng SQL:

|  |
| --- |
| select \* from Posts where Id = 1  select \* from Comments where PostId = 1 |

* + Để hiệu quả hơn chúng ta dùng join:

|  |
| --- |
| select \* from Posts  join Comments  on Posts.Id = Comments.Id  where Posts.Id = 1 |

* + Với DocDB, chúng ta có thể làm như sau

|  |
| --- |
| var post = docDb.Get<Post>(1);  var comments = docDb.Query<Comment>("CommentsByPost", query:"post\_id:1", orderBy:"+posted\_at"); |

Với index CommentsByPost được định nghĩa như sau

|  |
| --- |
| from doc in docs  where doc.type == "comment"  select new{doc.post\_id, doc.posted\_at} |

* + Lưu ý rằng chúng ta phải thực hiện 2 lời gọi ở đây bởi vì document database không có khái niệm join
* Post page: add comment to post
  + Sử dụng SQL:

|  |
| --- |
| insert into comments (PostId, ... )  values(1, ...) |

* + Sử dụng DocDB:

|  |
| --- |
| docDb.Store(new Comment{ PostId = 1, ... });  docDb.SaveChanges(); |

* Tag page: show all posts for tag
  + Sử dụng SQL thì hơi phức tạp một chút bởi vì tags có liên kết với blogs và posts. Chúng ta sẽ làm như sau:

|  |
| --- |
| select \* from Posts  where Id in (  select ItemId from tags  where ItemType = 'Posts' and TagId = 1  ) |

* + Sử dụng DocDB:

|  |
| --- |
| var posts = docDb.Query<Post>("PostsByTag", query:"tag:tags/1"); |

Với index được định nghĩa:

|  |
| --- |
| from doc in docs  where doc.type == "post"  from tag in doc.tags  select new {tag} |

* Categories page: show all posts for category: tương tự tags

Nhận xét: bằng cách sao chép mô hình dữ liệu cơ sở quan hệ, chúng ta đã tạo ra một số thứ mà chúng ta đã từng làm với cơ sở dữ liệu quan hệ. Và bây giờ chúng ta gặp phải vấn đề là document database không thể làm được những điều như cơ sở dữ liệu quan hệ có thể làm. Những gì chúng ta làm ở trên thật uổng phí. Chúng ta đạt được một số lợi ích nhỏ khi sử dụng schemaless (phi lược đồ), nhưng điều đó không thực sự có lợi so với những nỗ lực của chúng ta phải làm bằng cách cố gắng có được cơ sở dữ liệu quan hệ trên một cơ sở dữ liệu phi quan hệ.