**Link bài viết:** *https://viettelidc.com.vn/tin-tuc/so-sanh-2-loai-hinh-co-so-du-lieu-pho-bien-nhat-hien-nay-sql-va-nosql*

**SQL và NoSQL là gì?**

SQL (Structured Query Language), hay ngôn ngữ truy vấn có cấu trúc là một ngôn ngữ lập trình được dùng để tương tác với hệ thống quản lý [cơ sở dữ liệu quan hệ](https://viettelidc.com.vn/tin-tuc/co-so-du-lieu-quan-he-la-gi) (Oracle, MySQL, SQL Server, PostgreSQL…). Ngoài khả năng truy xuất, cập nhật và xóa bản ghi, SQL còn cho phép người dùng tạo và sửa đổi cấu trúc của database, cũng như kiểm soát các hoạt động truy cập.

Hệ thống SQL bao gồm nhiều thành phần như Database Engine, Integration Services, Notification Services, Analysis Services, Reporting Services, Service Broker… Mỗi thành phần đều cung cấp những công cụ cần thiết phục vụ cho việc quản lý và tối ưu hóa cơ sở dữ liệu. Nhờ khả năng bảo mật và truy vấn mạnh mẽ, SQL thường được tích hợp vào quy trình quản lý hệ thống, phân tích dữ liệu, phát triển web…

Bên cạnh SQL, các Developer còn sử dụng NoSQL (Not Only SQL/Non SQL) để xử lý dữ liệu phi cấu trúc với khối lượng lớn. NoSQL là loại database không sử dụng ngôn ngữ SQL để truy cập và tương tác dữ liệu. Khác với cơ sở dữ liệu quan hệ truyền thống, NoSQL lưu trữ, sắp xếp dữ liệu trong các mô hình Key-Value, Document, Graph, Wide-Column… Nhờ đó, NoSQL có thể mở rộng quy mô dễ dàng, hỗ trợ quản lý đa dạng nhiều loại dữ liệu mà không yêu cầu một định dạng cố định.

Bằng cách phân tán dữ liệu trên nhiều [máy chủ](https://viettelidc.com.vn/tin-tuc/may-chu-la-gi-dung-voi-bo-qua-5-tieu-chi-nay-khi-thue-may-chu-tu-cac-nha-cung-cap-3059), NoSQL giúp duy trì tính liên tục và sẵn sàng của hệ thống trong trường hợp phát sinh sự cố kỹ thuật. Đây là một lựa chọn lý tưởng dành cho những doanh nghiệp đang cần triển khai các ứng dụng web đòi hỏi tính linh hoạt và hiệu năng vượt trội.

**So sánh SQL và NoSQL**

Về điểm chung, cả SQL và NoSQL đều cho phép người dùng lưu trữ, quản lý và thực hiện các tác vụ đối với cơ sở dữ liệu. Tuy nhiên, giữa SQL và NoSQL vẫn có nhiều điểm khác biệt về cấu trúc, khả năng mở rộng, mô hình dữ liệu… Dưới đây là bảng so sánh chi tiết (mang tính tham khảo).

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **SQL** | **NoSQL** |
| Ngôn ngữ truy vấn | Sử dụng ngôn ngữ SQL để truy xuất dữ liệu. | Sử dụng ngôn ngữ truy vấn riêng, [API](https://viettelidc.com.vn/tin-tuc/api-la-gi-4-ung-dung-chinh-cua-api-trong-thiet-ke-website-ma-ban-nen-biet), công cụ truy vấn đặc thù thay cho SQL. |
| Khả năng mở rộng | Mở rộng theo chiều dọc bằng cách nâng cấp phần cứng. | Mở rộng theo chiều ngang bằng cách bổ sung node vào hệ thống. |
| Tính nhất quán | Tuân thủ các thuộc tính ACID (Atomicity, Consistency, Isolation, Durability), đảm bảo tính toàn vẹn và nhất quán của dữ liệu. | Áp dụng các thuộc tính BASE (Basically Available, Soft State, Eventually Consistent), ưu tiên tính sẵn sàng và khả năng chịu lỗi hơn tính nhất quán. |
| Cách thức lưu trữ | Dữ liệu được tổ chức theo mô hình xác định trước, bao gồm bảng, hàng và cột. | Dữ liệu có cấu trúc và phi cấu trúc có thể được lưu trữ linh hoạt dưới nhiều định dạng khác nhau. |
| Mô hình dữ liệu | Mô hình dữ liệu có cấu trúc (bảng). | Mô hình dữ liệu linh hoạt: Document, Graph, Key-Value, Wide-Column… |
| Phù hợp với ứng dụng nào | Phù hợp với những ứng dụng yêu cầu tính chính xác và mối quan hệ dữ liệu phức tạp (hệ thống tài chính, CRM, ERP…). | Phù hợp với những ứng dụng đòi hỏi xử lý khối lượng dữ liệu khổng lồ chỉ trong thời gian ngắn (ứng dụng web, di động, thiết bị [IoT](https://viettelidc.com.vn/tin-tuc/iot-la-gi-nhung-ung-dung-noi-bat-cua-iot)…). |
| Ví dụ | Oracle, MySQL, MariaDB, SQL Server… | MongoDB, Couchbase, Cassandra, Redis… |

**Khi nào chọn SQL? Khi nào chọn NoSQL?**

Doanh nghiệp nên sử dụng ngôn ngữ SQL khi cần:

- Quản lý, thực hiện thao tác chỉnh sửa, truy xuất, thêm/xóa dữ liệu có cấu trúc trong các cơ sở dữ liệu quan hệ.

- Duy trì tính toàn vẹn, đồng nhất của dữ liệu để tránh xảy ra sai sót trong quá trình read-write.

- Báo cáo và trực quan hóa dữ liệu.

- Phân tích dữ liệu với khối lượng vừa và lớn.

Ngoài ra, doanh nghiệp cũng có thể cân nhắc lựa chọn NoSQL thay thế cho cơ sở dữ liệu quan hệ truyền thống trong những trường hợp dưới đây:

- Quản lý dữ liệu phi cấu trúc hoặc có cấu trúc thay đổi theo thời gian, chẳng hạn như dữ liệu cảm biến từ thiết bị IoT, dữ liệu văn bản, hình ảnh…

- Xây dựng hệ thống yêu cầu tính sẵn sàng cao, ví dụ: ứng dụng ngân hàng, ứng dụng y tế…

- Cần duy trì hiệu suất ổn định và liên tục trong quá trình truy xuất dữ liệu.

- Cần mở rộng hệ thống nhanh chóng để đáp ứng nhu cầu lưu trữ khối lượng dữ liệu khổng lồ, phù hợp với quy mô tăng trưởng của doanh nghiệp.

# **Link bài viết:** [*https://itviec.com/blog/sql-vs-nosql/*](https://itviec.com/blog/sql-vs-nosql/)

# **SQL vs NoSQL: Cách chọn hệ quản trị cơ sở dữ liệu phù hợp**

***Trong thế giới cơ sở dữ liệu, SQL database và NoSQL database là hai lựa chọn phổ biến và được sử dụng rộng rãi. Mỗi loại cơ sở dữ liệu đều có những đặc trưng và ưu điểm riêng, phù hợp với những trường hợp khác nhau. Việc hiểu rõ sự khác biệt giữa SQL vs NoSQL sẽ giúp bạn lựa chọn hệ thống lưu trữ dữ liệu phù hợp với nhu cầu cụ thể, tối ưu hóa hiệu suất và dễ dàng quản lý.***

Đọc bài viết này để hiểu rõ hơn về:

* Tổng quan về SQL
* Tổng quan về NoSQL
* Những điểm khác biệt giữa SQL vs NoSQL
* Những trường hợp sử dụng SQL vs NoSQL

## SQL là gì?

SQL (Structured Query Language) là ngôn ngữ tiêu chuẩn được sử dụng để tương tác với các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) hoặc hệ quản trị cơ sở dữ liệu quan hệ – đối tượng (ORDBMS).

SQL cho phép người dùng thực hiện các thao tác như truy vấn, chèn, cập nhật và xóa dữ liệu, đồng thời cung cấp các tính năng quản lý và tối ưu hóa cơ sở dữ liệu để đảm bảo tính nhất quán và toàn vẹn dữ liệu. Với cú pháp đơn giản và dễ hiểu, SQL được sử dụng rộng rãi trong nhiều lĩnh vực như phân tích dữ liệu, phát triển phần mềm, và quản lý hệ thống.

Tham khảo chia sẻ độc quyền giữa Chuyên gia PL/SQL Developer với 10+ năm kinh nghiệm với ITviec: [**Tất tần tật về ngôn ngữ SQL đều nằm trong bài viết này**](https://itviec.com/blog/sql-la-gi/)

### **Dữ liệu có cấu trúc là gì?**

Dữ liệu có cấu trúc (structure data) là dạng dữ liệu mà các thông tin được tổ chức và định dạng một cách rõ ràng, theo một khuôn mẫu nhất định.

Trong cơ sở dữ liệu quan hệ, dữ liệu có cấu trúc được lưu trữ dưới dạng bảng (table). Các bảng này có thể được liên kết với nhau thông qua các mối quan hệ (relationships) nhờ vào các khóa (key) như *PRIMARY KEY* và *FOREIGN KEY*. Nhờ đó cho phép việc lưu trữ và quản lý dữ liệu một cách có tổ chức, giúp dễ dàng truy vấn và duy trì tính toàn vẹn của thông tin.

### **SQL database là gì?**

SQL database là hệ quản trị cơ sở dữ liệu (DBMS) sử dụng ngôn ngữ SQL để quản lý và tương tác với dữ liệu được tổ chức dưới dạng các bảng có quan hệ. Các APIs (giao diện lập trình ứng dụng) của SQL cung cấp nhiều nhóm hàm giúp nhà phát triển có thể thực hiện các thao tác với cơ sở dữ liệu một cách hiệu quả mà không cần phải lặp đi lặp lại từng câu lệnh SQL.

Một số hệ quản trị SQL phổ biến bao gồm MySQL, SQL Server và PostgreSQL. Các SQL database hỗ trợ người dùng định nghĩa cấu trúc dữ liệu, thực hiện các truy vấn phức tạp và tối ưu hóa hiệu suất thông qua các tính năng như *index*, *JOIN*, *stored procedures*…

Ngoài ra bạn có thể tìm hiểu thêm về [**SQL Database**](https://itviec.com/blog/sql-database-la-gi/).

### **Ưu điểm của SQL**

* Cung cấp bộ lệnh chuẩn hóa: SQL sở hữu một bộ lệnh chuẩn cho việc định nghĩa, truy vấn, cập nhật và quản trị dữ liệu trong hệ quản trị cơ sở dữ liệu quan hệ (RDBMS). Các lệnh cơ bản bao gồm: SELECT, INSERT, DELETE, UPDATE,…
* Khả năng thực hiện các truy vấn phức tạp: SQL hỗ trợ các truy vấn phức tạp bằng cách sử dụng JOIN, GROUP BY, ORDER BY và các hàm tính toán, tổng hợp, giúp cho việc  phân tích dữ liệu và trích xuất thông tin từ nhiều bảng trong cơ sở dữ liệu dễ dàng hơn.
* Độ tin cậy cao: Nhờ tuân thủ theo tính chất ACID, các hệ thống SQL đảm bảo dữ liệu luôn được toàn vẹn, nhất quán và bền vững ngay cả khi xảy ra lỗi hay sự cố bất ngờ.
* Bảo mật cao: SQL cung cấp cơ chế bảo mật mạnh mẽ như kiểm soát truy cập và quyền hạn chi tiết (Role-Based Access Control – RBAC), cho phép quản trị viên xác định chính xác ai có thể xem, chỉnh sửa hay xóa dữ liệu. Mã hóa dữ liệu giúp bảo vệ dữ liệu khỏi những truy cập trái phép.
* Tương thích với nhiều ứng dụng và hệ thống: Do SQL đã tồn tại và phát triển lâu đời, nó có khả năng tương thích tốt với nhiều công cụ BI (Business Intelligence), phần mềm phân tích và các ứng dụng kinh doanh. SQL cũng có khả năng kết nối với các ngôn ngữ lập trình phổ biến như Python, Java, PHP và các framework như Django, Ruby on Rails.

### **Nhược điểm của SQL**

* Giao diện phức tạp: Mặc dù SQL dễ học cho các thao tác cơ bản, nhưng việc sử dụng các tính năng nâng cao và tối ưu hóa hiệu suất có thể phức tạp, đòi hỏi kỹ năng chuyên sâu từ quản trị viên cơ sở dữ liệu.
* Giới hạn về khả năng mở rộng với dữ liệu lớn: Khi xử lý các tập dữ liệu lớn và phức tạp, hiệu suất của SQL có thể giảm, đặc biệt là đối với các truy vấn phức tạp hoặc cơ sở dữ liệu không được tối ưu hóa.
* Thiếu linh hoạt cho dữ liệu phi cấu trúc: SQL được thiết kế cho dữ liệu có cấu trúc (tables, columns), nên gặp khó khăn khi xử lý dữ liệu phi cấu trúc hoặc bán cấu trúc.
* Vấn đề bảo mật tiềm ẩn: Mặc dù SQL có các tính năng bảo mật mạnh mẽ, nhưng nó vẫn dễ bị tấn công SQL Injection nếu ứng dụng không được mã hóa cẩn thận, dẫn đến nguy cơ xâm nhập trái phép hoặc rò rỉ dữ liệu.
* Chi phí triển khai và bảo trì cao: Việc triển khai và duy trì hệ thống SQL mạnh mẽ, đặc biệt là với các hệ quản trị cơ sở dữ liệu thương mại, có thể tốn kém. Các doanh nghiệp nhỏ có thể gặp khó khăn trong việc đáp ứng chi phí này.

## NoSQL là gì?

NoSQL (Not only SQL) là một thuật ngữ chung chỉ các cơ sở dữ liệu không dựa trên mô hình quan hệ truyền thống như các cơ sở dữ liệu SQL.

Thay vì tập trung vào việc duy trì các mối quan hệ giữa các bảng, NoSQL được thiết kế để xử lý các tập dữ liệu lớn và phức tạp, giải quyết các bài toán về khả năng mở rộng linh hoạt mà các cơ sở dữ liệu quan hệ (RDBMS) truyền thống không thể đáp ứng hiệu quả.

NoSQL cho phép lưu trữ dữ liệu dưới nhiều định dạng khác nhau như key-value, document, graph, và column-family, giúp tối ưu hóa việc lưu trữ, truy xuất và xử lý dữ liệu phi cấu trúc hoặc bán cấu trúc.

Đọc thêm: [***NoSQL là gì? Vì sao gọi NoSQL là bước tiến mới trong quản lý dữ liệu?***](https://itviec.com/blog/nosql-la-gi/)

### **Dữ liệu phi cấu trúc là gì?**

Dữ liệu phi cấu trúc là dạng dữ liệu không tuân theo một mô hình dữ liệu dạng bảng truyền thống, không có cấu trúc rõ ràng và dễ nhận biết như trong cơ sở dữ liệu quan hệ. Các thông tin như tài liệu văn bản, hình ảnh, video hoặc các bài đăng trên mạng xã hội thường có định dạng khác nhau và không có cấu trúc rõ ràng.

Dữ liệu phi cấu trúc khó xử lý và tổ chức theo dạng bảng truyền thống, nhưng lại phù hợp với cách lưu trữ của NoSQL, nơi mà mỗi tài liệu hoặc đối tượng có thể chứa dữ liệu theo nhiều dạng và kiểu khác nhau.

### **NoSQL database là gì?**

NoSQL database là hệ quản trị cơ sở dữ liệu hỗ trợ lưu trữ và truy xuất dữ liệu phi cấu trúc, bán cấu trúc và cả dữ liệu có cấu trúc mà không cần dựa trên các bảng quan hệ. NoSQL database chia thành nhiều loại với các đặc điểm khác nhau để phù hợp với từng loại dữ liệu và yêu cầu xử lý:

* Document databases: Lưu trữ dữ liệu dưới dạng tài liệu JSON hoặc BSON. Mỗi tài liệu có thể chứa các trường (field) khác nhau, linh hoạt trong việc tổ chức dữ liệu. Ví dụ: MongoDB, Couchbase.
* Key-value databases: Lưu trữ dữ liệu theo cặp key-value, cho phép truy xuất dữ liệu nhanh chóng thông qua khóa. Đây là dạng mô hình đơn giản nhất trong hệ NoSQL, nó phù hợp cho các hệ thống cần tìm kiếm nhanh, xử lý tốc độ cao như bộ nhớ đệm (caching) hay quản lý session. Ví dụ: Redis, DynamoDB.
* Column-family stores: Tổ chức dữ liệu theo cột thay vì hàng như trong RDBMS. Column-family stores có khả năng tối ưu hóa cho việc xử lý truy vấn nên rất phù hợp cho các ứng dụng phân tích dữ liệu lớn và các ứng dụng đòi hỏi hiệu suất cao. Ví dụ: Cassandra, HBase.
* Graph databases: Lưu trữ dữ liệu dưới dạng các nút và mối quan hệ (edges), giúp thể hiện và xử lý dữ liệu phức tạp có liên kết mạnh mẽ, phù hợp với các ứng dụng như mạng xã hội, hệ thống đề xuất. Ví dụ: Neo4j, ArangoDB.

Đọc thêm: [***Các loại cơ sở dữ liệu NoSQL: Định nghĩa, Ưu – Nhược điểm và Ứng dụng***](https://itviec.com/blog/cac-loai-co-so-du-lieu-nosql/)

### **Ưu điểm của NoSQL**

* Khả năng mở rộng theo chiều ngang: NoSQL được thiết kế để mở rộng theo chiều ngang, tức là có thể thêm nhiều máy chủ hơn vào hệ thống phân tán để xử lý dữ liệu lớn và dung lượng lưu trữ. Điều này khác với SQL khi thường yêu cầu mở rộng theo chiều dọc (nâng cấp phần cứng hệ thống).
* Cân bằng tải tự động và phân mảnh (Sharding): Nhiều cơ sở dữ liệu NoSQL cung cấp tính năng phân mảnh và cân bằng tải tự động, giúp phân phối dữ liệu trên nhiều nút trong hệ thống, từ đó cải thiện hiệu suất và sử dụng tài nguyên hiệu quả hơn.
* Linh hoạt trong cấu trúc dữ liệu: NoSQL sử dụng lược đồ động, cho phép thay đổi cấu trúc dữ liệu một cách linh hoạt. Các trường trong một bản ghi có thể khác nhau giữa các tài liệu, giúp dễ dàng thích nghi với những thay đổi hoặc dữ liệu không đồng nhất.
* Xử lý tốt dữ liệu phi cấu trúc và bán cấu trúc: NoSQL tối ưu cho việc lưu trữ các loại dữ liệu phi cấu trúc hoặc bán cấu trúc như mạng xã hội, video, hình ảnh,… và thường được sử dụng trong các ứng dụng hiện đại với nguồn dữ liệu không ổn định và khó đoán trước (ví dụ như dữ liệu do người dùng tạo ra).
* Hiệu suất cao: Khả năng truy xuất và xử lý dữ liệu nhanh chóng nhờ vào cách tổ chức lưu trữ tối ưu cho từng loại dữ liệu.
* Dễ dàng tích hợp với các công nghệ hiện đại: NoSQL dễ dàng tích hợp với các công nghệ tiên tiến như microservices, cloud computing và big data, giúp NoSQL trở thành lựa chọn lý tưởng cho các ứng dụng hiện đại, nơi mà yêu cầu về tính linh hoạt, mở rộng và hiệu suất cao là rất cần thiết.

### **Nhược điểm của NoSQL**

* Thiếu chuẩn hóa: Mỗi loại cơ sở dữ liệu NoSQL có cách tổ chức và truy vấn dữ liệu riêng và không có ngôn ngữ truy vấn chuẩn như SQL, gây khó khăn khi di chuyển hoặc làm việc với các hệ thống NoSQL khác nhau.
* Hỗ trợ ACID hạn chế: NoSQL thường không hoàn toàn tuân thủ các thuộc tính ACID (Tính nguyên tử, Nhất quán, Cô lập và Bền vững). Điều này có thể ảnh hưởng đến tính toàn vẹn của dữ liệu trong các ứng dụng đòi hỏi sự chính xác cao về giao dịch.
* Khả năng giao dịch hạn chế: Các hệ thống NoSQL thiếu hỗ trợ đầy đủ cho các giao dịch phức tạp, làm giảm khả năng thực hiện các hoạt động dữ liệu phức tạp và quan trọng trong một số trường hợp.
* Thách thức trong quản lý tính nhất quán: NoSQL ưu tiên khả năng phân vùng và tính sẵn sàng, dẫn đến các mô hình nhất quán cuối cùng (eventual consistency). Điều này có thể gây khó khăn trong việc đảm bảo tính nhất quán của dữ liệu trong một số trường hợp.
* Cộng đồng và công cụ hỗ trợ hạn chế: NoSQL thường có hệ sinh thái và tài nguyên hỗ trợ ít phát triển hơn so với SQL. Điều này có thể làm tăng khó khăn cho người dùng trong việc tìm kiếm giải pháp hoặc xử lý sự cố khi làm việc với NoSQL.
* Phức tạp trong việc mở rộng: Mặc dù NoSQL hỗ trợ mở rộng theo chiều ngang, việc quản lý hệ thống phân tán, phân vùng dữ liệu và đảm bảo tính sẵn sàng cao có thể tạo ra những thách thức phức tạp cho quản trị viên.

## So sánh tổng quan SQL vs NoSQL

|  |  |  |
| --- | --- | --- |
| Tiêu chí | SQL | NoSQL |
| Loại CSDL | Cơ sở dữ liệu quan hệ (Relational Database – RDBMS). | Cơ sở dữ liệu phi quan hệ (Non-relational Database). |
| Ngôn ngữ truy vấn | Sử dụng Structured Query Language (SQL), ngôn ngữ mạnh mẽ và phổ biến | Không có ngôn ngữ truy vấn cố định, cấu trúc truy vấn tùy thuộc vào từng loại NoSQL |
| Mô hình lưu trữ dữ liệu | Lưu trữ dữ liệu trong bảng, bao gồm cột và hàng. | Lưu trữ dữ liệu ở nhiều dạng như document, key-value, cột (column family) hoặc đồ thị. |
| Xử lý loại dữ liệu | Xử lý dữ liệu có cấu trúc. | Có thể xử lý dữ liệu có cấu trúc, bán cấu trúc hoặc phi cấu trúc, linh hoạt với nhiều loại dữ liệu khác nhau. |
| Schema (lược đồ) | Có schema cố định, yêu cầu phải tuân theo cấu trúc định sẵn. | Sử dụng schema động, linh hoạt cho phép thêm dữ liệu mà không cần cấu trúc định trước. |
| Hỗ trợ truy vấn | SQL hỗ trợ các truy vấn phức tạp và phép JOIN giữa các bảng. | Một số loại NoSQL không hỗ trợ phép JOIN và thường khó khăn trong việc thực hiện các truy vấn phức tạp. |
| Khả năng mở rộng | Mở rộng theo chiều dọc (vertical scaling) bằng cách nâng cấp hệ thống. | Mở rộng theo chiều ngang (horizontal scaling) bằng cách thêm nhiều máy chủ hoặc node. |
| Tính nhất quán giao dịch | Tuân theo nguyên lý ACID (Atomicity, Consistency, Isolation, Durability), giúp đảm bảo tính nhất quán trong giao dịch. Tuy nhiên, không phải lúc nào cũng đảm bảo “nhất quán tức thì” trong mọi hệ thống, vì điều này còn phụ thuộc vào cách cơ sở dữ liệu được cấu hình và triển khai. “Nhất quán tức thì” có nghĩa là mọi thay đổi trong giao dịch sẽ được đưa vào cơ sở dữ liệu một cách hợp lệ từ trạng thái này sang trạng thái khác, đảm bảo rằng dữ liệu luôn đồng nhất trên toàn hệ thống | Sử dụng “eventual consistency” — có nghĩa là trong một hệ thống phân tán, sự thay đổi dữ liệu không được đồng bộ ngay lập tức trên tất cả các node, nhưng sau một khoảng thời gian, toàn bộ hệ thống sẽ dần đồng nhất về mặt dữ liệu. Điều này giúp tăng khả năng mở rộng nhưng có thể dẫn đến tình trạng không nhất quán tạm thời |
| Hiệu suất với khối lượng dữ liệu lớn | Khi khối lượng dữ liệu lớn, SQL có thể gặp hạn chế về hiệu suất do việc mở rộng theo chiều dọc thường có chi phí cao và khó duy trì | NoSQL với khả năng mở rộng theo chiều ngang cho phép xử lý tốt hơn khi khối lượng dữ liệu lớn và có thể xử lý các ứng dụng lớn một cách hiệu quả hơn |

## SQL vs NoSQL: Khi nào nên sử dụng SQL?

SQL nên được sử dụng khi:

* Khi dữ liệu có cấu trúc rõ ràng và ổn định.
* Khi yêu cầu tính toàn vẹn và nhất quán của dữ liệu cao, cần đảm bảo dữ liệu được cập nhật và phản ánh ngay lập tức sau các giao dịch.
* Khi cần quản lý các mối quan hệ phức tạp giữa các bảng dữ liệu.
* Khi hệ thống dữ liệu không bị thay đổi thường xuyên, lượng dữ liệu lưu trữ không quá lớn.
* Khi những ứng dụng cần yêu cầu tính năng giao dịch (transaction).

Sau đây là một số use case cụ thể cho SQL:

* Hệ thống quản lý ngân hàng: Ngân hàng cần xử lý giao dịch tài chính như gửi tiền, rút tiền, hoặc chuyển khoản với độ chính xác cao và tức thì. SQL đảm bảo các giao dịch này tuân theo nguyên tắc ACID, giúp bảo vệ dữ liệu khỏi sự mất mát hay sai lệch.
* Ứng dụng bán lẻ trực tuyến (eCommerce): Các trang bán hàng trực tuyến như Amazon cần theo dõi sản phẩm, khách hàng, đơn hàng, và giao dịch. SQL giúp đảm bảo tính nhất quán khi sản phẩm được thêm vào giỏ hàng và trừ kho tương ứng.
* Quản lý hệ thống khách hàng (CRM): Các nền tảng như Salesforce sử dụng SQL để quản lý dữ liệu khách hàng, mối quan hệ với các đơn hàng, dịch vụ. Mối quan hệ phức tạp giữa khách hàng và dữ liệu giao dịch được SQL xử lý tốt nhờ cấu trúc bảng và các khóa ngoại (foreign key).

## SQL vs NoSQL: Khi nào nên sử dụng NoSQL?

NoSQL nên được sử dụng khi:

* Khi dữ liệu không có cấu trúc hoặc cấu trúc có thể thay đổi liên tục như JSON, XML, hình ảnh, video, các bài đăng mạng xã hội.
* Khi cần xử lý khối lượng dữ liệu lớn và mở rộng hệ thống một cách dễ dàng bằng cách thêm nhiều máy chủ (mở rộng ngang).
* Khi hiệu suất và khả năng mở rộng là yếu tố chính, đặc biệt trong các hệ thống phân tích dữ liệu lớn thời gian thực.

Sau đây là một số use case cụ thể cho NoSQL:

* Mạng xã hội: Ví dụ như Facebook, Twitter phải xử lý khối lượng dữ liệu lớn không có cấu trúc từ bài viết, bình luận đến các hành vi người dùng như lượt thích, chia sẻ. NoSQL cho phép lưu trữ và truy xuất dữ liệu một cách linh hoạt mà không cần cấu trúc cố định.
* Ứng dụng phân tích dữ liệu lớn thời gian thực: Các nền tảng như Splunk hoặc Hadoop sử dụng NoSQL để xử lý và phân tích hàng triệu sự kiện thời gian thực, từ log hệ thống đến dữ liệu từ cảm biến. Điều này giúp đưa ra các quyết định kịp thời dựa trên dữ liệu không có cấu trúc.
* Ứng dụng game online: Các trò chơi trực tuyến cần lưu trữ và xử lý thông tin người chơi, điểm số và các sự kiện trong game. NoSQL hỗ trợ việc mở rộng hệ thống khi có hàng triệu người chơi truy cập đồng thời, đảm bảo hiệu năng và khả năng mở rộng.

## *Các câu hỏi thường gặp về SQL vs NoSQL*

### **SQL vs NoSQL: NoSQL có nhanh hơn SQL không?**

Không thể khẳng định rằng NoSQL nhanh hơn SQL hay ngược lại vì tốc độ của cả hai phụ thuộc vào ngữ cảnh sử dụng.

SQL được thiết kế tối ưu cho các truy vấn, kết nối (joins) và cập nhật dữ liệu, đặc biệt khi cần tránh trùng lặp dữ liệu và đảm bảo tính toàn vẹn. Điều này khiến SQL nhanh hơn trong các tác vụ liên quan đến truy vấn phức tạp.

Ngược lại, NoSQL lưu trữ dữ liệu cùng nhau, không tách rời như SQL, giúp cho việc đọc và ghi dữ liệu trên một thực thể nhanh hơn trong các tình huống với dữ liệu không có cấu trúc và yêu cầu xử lý khối lượng lớn dữ liệu. Về cơ bản, SQL tối ưu khi cần bảo vệ tính chính xác của dữ liệu, còn NoSQL phù hợp hơn khi cần tốc độ và tính sẵn sàng cao với dữ liệu lớn.

### **SQL vs NoSQL: SQL database có thể xử lý cùng loại dữ liệu như NoSQL database không?**

Cơ sở dữ liệu SQL được thiết kế để xử lý dữ liệu có cấu trúc và rất hiệu quả trong việc quản lý các truy vấn phức tạp và giao dịch với dữ liệu quan hệ. Mặc dù SQL có thể xử lý dữ liệu có cấu trúc một cách hiệu quả, nhưng nó không linh hoạt bằng NoSQL trong việc xử lý dữ liệu không có cấu trúc hoặc bán cấu trúc do yêu cầu về lược đồ cố định.

### **SQL vs NoSQL: NoSQL thường được ưa chuộng hơn SQL khi nào?**

Trong một số trường hợp nhất định, NoSQL database thường được xem là lựa chọn ưa thích hơn thay vì SQL database vì những lý do như:

* Khả năng mở rộng theo chiều ngang: Đối với những ứng dụng lớn, thường đối mặt với khối lượng dữ liệu khổng lồ thì việc mở rộng theo chiều ngang sẽ tối ưu hơn so với mở rộng theo chiều dọc như SQL.
* Truy vấn đơn giản nhưng yêu cầu hiệu suất cao: Những ứng dụng yêu cầu cao về tốc độ sẽ ưu tiên sử dụng NoSQL hơn.
* Xử lý dữ liệu phi cấu trúc: Những ứng dụng có nhiều nguồn dữ liệu khác nhau và đặc thù như hình ảnh, video, JSON,…
* Cân bằng tải tự động và hiệu quả cao trong hệ thống phân tán: NoSQL hỗ trợ tính năng phân mảnh dữ liệu (sharding) và cân bằng tải tự động, giúp phân phối dữ liệu và truy vấn hiệu quả trên nhiều máy chủ, từ đó cải thiện hiệu suất và đảm bảo sự ổn định của hệ thống.

## Tổng kết SQL vs NoSQL

Việc lựa chọn giữa SQL vs NoSQL phụ thuộc nhiều vào yêu cầu của dự án và đặc điểm dữ liệu mà bạn xử lý. SQL thường là lựa chọn lý tưởng khi dữ liệu có cấu trúc rõ ràng và tính nhất quán cao, trong khi NoSQL lại mang đến sự linh hoạt và khả năng mở rộng tốt hơn cho các ứng dụng hiện đại với dữ liệu phi cấu trúc hoặc bán cấu trúc.

## Link bài viết: [*https://bkhost.vn/blog/so-sanh-sql-va-nosql/*](https://bkhost.vn/blog/so-sanh-sql-va-nosql/)

## So sánh SQL và NoSQL

### Định nghĩa và Khái niệm

#### SQL (Structured Query Language)

**SQL**, còn được gọi là Ngôn ngữ truy vấn có cấu trúc, là một ngôn ngữ lập trình được sử dụng để truy vấn, chèn, cập nhật và xóa dữ liệu từ các hệ quản trị cơ sở dữ liệu quan hệ. SQL dựa trên mô hình dữ liệu bảng và quan hệ giữa các bảng. Đây là một ngôn ngữ mạnh mẽ và phổ biến được sử dụng trong các hệ thống dựa trên mô hình quan hệ.

#### NoSQL (Non-SQL hoặc Not Only SQL)

**NoSQL** là một loại hệ quản trị cơ sở dữ liệu không sử dụng ngôn ngữ truy vấn có cấu trúc như SQL. Thay vào đó, NoSQL sử dụng các mô hình dữ liệu khác nhau như cơ sở dữ liệu cột, tài liệu, đồ thị hoặc key-value. Mục tiêu của NoSQL là giải quyết những hạn chế của hệ quản trị cơ sở dữ liệu quan hệ và hỗ trợ quy mô lớn và tính mở rộng tốt hơn.

### Các Loại Cơ Sở Dữ Liệu

#### SQL

* Hệ quản trị cơ sở dữ liệu SQL dựa trên mô hình quan hệ và sử dụng bảng để lưu trữ dữ liệu.
* Các bảng có các hàng và cột, và mỗi hàng đại diện cho một bản ghi trong cơ sở dữ liệu.
* SQL hỗ trợ các ràng buộc (constraints) như khóa chính, ràng buộc duy nhất và ràng buộc ngoại.
* Các hệ quản trị cơ sở dữ liệu SQL phổ biến bao gồm MySQL, [PostgreSQL](https://bkhost.vn/blog/postgresql/) và Oracle.

#### NoSQL

* NoSQL hỗ trợ nhiều mô hình dữ liệu, bao gồm cơ sở dữ liệu cột (column-family), cơ sở dữ liệu tài liệu (document-based), cơ sở dữ liệu đồ thị (graph-based) và cơ sở dữ liệu key-value.
* Mỗi loại cơ sở dữ liệu NoSQL có đặc điểm riêng và thích hợp với các ứng dụng cụ thể.
* Các hệ quản trị cơ sở dữ liệu NoSQL phổ biến bao gồm [MongoDB](https://bkhost.vn/blog/mongodb/), Cassandra và Neo4j.

### Ưu Nhược Điểm

#### SQL

**Ưu điểm**

* Hỗ trợ [ACID](https://bkhost.vn/blog/acid-transactions/) (Atomicity, Consistency, Isolation, Durability), đảm bảo tính toàn vẹn và nhất quán của dữ liệu.
* Truy vấn dữ liệu dễ dàng và linh hoạt với ngôn ngữ truy vấn SQL chuẩn.
* Dữ liệu có cấu trúc, giúp giảm thiểu việc lặp lại thông tin và tối ưu hóa hiệu suất truy vấn.

**Nhược điểm**

* Hạn chế trong việc mở rộng quy mô lớn và xử lý dữ liệu phi cấu trúc.
* Khó khăn trong việc thay đổi cấu trúc dữ liệu khi ứng dụng phát triển.

#### NoSQL

**Ưu điểm**

* Hỗ trợ quy mô lớn và dễ dàng mở rộng hệ thống.
* Được thiết kế để xử lý dữ liệu phi cấu trúc và linh hoạt trong việc thay đổi cấu trúc dữ liệu.
* Hiệu suất cao khi truy vấn dữ liệu lớn và phức tạp.

**Nhược điểm**

* Không hỗ trợ đầy đủ các tính năng ACID, có thể dẫn đến sự mất mát dữ liệu trong một số tình huống.
* Cú pháp truy vấn và tương tác với dữ liệu phức tạp hơn so với SQL.

## Lựa Chọn Hệ Quản Trị Cơ Sở Dữ Liệu Phù Hợp

Trước khi quyết định sử dụng SQL hay NoSQL, hãy xem xét các yếu tố sau:

* **Kiểu dữ liệu**: Nếu dữ liệu của bạn có cấu trúc rõ ràng và bạn cần tính toàn vẹn và nhất quán, SQL là lựa chọn tốt. Tuy nhiên, nếu dữ liệu của bạn không có cấu trúc rõ ràng và bạn cần mở rộng hệ thống, NoSQL có thể là lựa chọn hợp lý.
* **Yêu cầu về quy mô**: Nếu bạn dự định xây dựng một ứng dụng với quy mô lớn và cần mở rộng hệ thống dễ dàng, NoSQL là sự lựa chọn đáng xem. SQL thường hạn chế trong việc mở rộng quy mô lớn.
* **Khả năng thay đổi**: Nếu ứng dụng của bạn đang phát triển và dự kiến ​​có nhiều thay đổi trong cấu trúc dữ liệu, NoSQL có thể giúp bạn dễ dàng thay đổi mô hình dữ liệu.
* **Hiệu suất**: Tùy thuộc vào yêu cầu hiệu suất của ứng dụng, bạn có thể lựa chọn SQL hoặc NoSQL. NoSQL thường có hiệu suất tốt hơn khi xử lý lượng dữ liệu lớn và phức tạp.

## Kết Luận

SQL và NoSQL đều có ưu nhược điểm riêng và thích hợp với các trường hợp sử dụng cụ thể. Để lựa chọn hệ quản trị cơ sở dữ liệu phù hợp, hãy xem xét cẩn thận yêu cầu của ứng dụng và tính năng mà bạn cần. Nếu bạn cần tính toàn vẹn dữ liệu và quy mô không quá lớn, SQL có thể là lựa chọn tốt. Nếu bạn cần mở rộng hệ thống và xử lý dữ liệu phi cấu trúc, hãy xem xét sử dụng NoSQL.