Database

Mô hình cơ sở dữ liệu: SQL hoặc No-SQL:

SQL

1. You’re working with complex queries and reports. With SQL you can build one script that retrieves and presents your data. NoSQL doesn’t support relations between data types. Running queries in NoSQL is doable, but much slower.
2. You have a high transaction application. SQL databases are a better fit for heavy duty or complex transactions because it’s more stable and ensure data integrity.
3. You need to ensure ACID compliance (Atomicity, Consistency, Isolation, Durability) or defining exactly how transactions interact with a database.
4. You don’t anticipate a lot of changes or growth. If you’re not working with a large volume of data or many data types, NoSQL would be overkill.

No-SQL

1. You are constantly adding new features, functions, data types, and it’s difficult to predict how the application will grow over time.
2. Changing a data model is SQL is clunky and requires code changes. A lot of time is invested designing the data model because changes will impact all or most of the layers in the application.
3. In NoSQL, we are working with a highly flexible schema design or no predefined schema. The data modeling process is iterative and adaptive. Changing the structure or schema will not impact development cycles or create any downtime for the application.
4. You are not concerned about data consistency and 100% data integrity is not your top goal. This is related to the above SQL requirement for ACID compliance. For example, with social media platforms, it isn’t important if everyone sees your new post at the exact same time, which means data consistency is not a priority.
5. You have a lot of data, many different data types, and your data needs will only grow over time. NoSQL makes it easy to store all different types of data together and without having to invest time into defining what type of data you’re storing in advance.
6. Your data needs scale up, out, and down. As discussed above, NoSQL provides much greater flexibility and the ability to control costs as your data needs change.

Tài liệu tham khảo [When to Use SQL vs. NoSQL | Integrant](https://integrant.com/blog/when-to-use-sql-vs-nosql)

Vì các tính chất của nghiệp vụ cần:

* Xuất báo cáo
* Đảm bảo tính toàn vẹn, tính nhất quán, giảm dư thừa dữ liệu
* Các tính năng và kiểu dự liệu có thể dự đoán trước
* Ít có sự thay đổi lớn về nghiệp vụ khi hệ thống dần đi vào ổn định
* Không cần dữ liệu phân tán

=> Chọn cơ sở dữ liệu quan hệ.

Chấp nhận nhược điểm tồn tại:

* Khó khăn trong việc thiết kế
* Khó khăn khi gặp sự thay đổi nhất định

MySQL, PostgreSQL, SQL-server

Đây là 3 SQL Data Base được sử dụng nhiều nhất hiện tại

SQL-server cần phải trả phí, không phù hợp tài nguyên của đội ngũ phát triển -> bỏ

Chọn PostgreSQL vì PostgreSQL có những nhiều ưu điểm so với MySQL:

* PostgreSQL được dự đoán sẽ chiếm vị trí SQL Database phổ biến nhất của MySQL trong vòng 3 năm nữa
* Hỗ trợ Analytic functions
* Ngoài các Data types chuẩn sql thì còn có array, hstore, object (người dùng định nghĩa)
* Full text search: không giới hạn
* Thuật toán JOIN sử lí tối ưu hơn
* DELETE nhanh hơn

Chấp nhận nhược điểm:

* UPDATE chậm hơn MySQL
* MySQL SELECT TOP nhanh hơn

Front-end:   
Ngôn ngữ: java-script vì nó phổ biến, support mạnh, vì làm trên cả web và di động nên chọn java-script để tái sử dụng code giữa các nền tảng

Plat form: Native và Cross ?: chọn Cross vì nghiệp vụ không cần can thiệp sâu vào phần cứng và viết code 1 lần build trên nhiều hệ điều hành

Cụ thể chọn Expo: frame word cho phép code 1 lần build web, android, ios (dĩ nhiên phải custom responsive và 1 số sự khác biệt không có thư viện không dùng chung)

Nhược điểm: sản phẩm chậm hơn Naïve form và web truyền thống, source lẫn lộn chung

Ưu: Code ít, tái sử dụng tốt, phù hợp trong việc cho ra sản phẩm nhanh mà ít nhân lực

Back-end:

Java Spring-boot