|  |  |
| --- | --- |
| SQL | NOSQL |
| Nesting correlation between different tables is done through joins and foreign keys | Nesting is done via referring objected, this is fast as we are just referencing and not adding document to database |
| There are predefined schemas for each table you will be creating | Mongo is flexible, it makes it easier and faster to change the data model , but this increases risk of having inconsistent data |
| Consistency is assured, hence inserting is slower | Consistency is not as much hence inserting is faster as the additional checks for consistency is not that strict . Performance is good |
| ACID property is assured | Not |
| Most used for transactional based data | Used for inventory based ecommerce websites where each object has horizontal scaling such as one shirt can have diff colour diff sizes etc |
| **INSERTION 🡪 slower** | **Faster** |
| **Accesing 🡪 Faster** | **Slower (specially when searching for an attribute that is not indexed)** |
| **Sorting🡪 Faster** | **Slower ( Mongo is slower than table when it comes to fetching value inside a doc)** |
| **Joins 🡪 slower** | **Faster ( you can avoid joins with nested documents** |
| **Updating🡪slower because it has to change the foreign key constraints** | **Faster (you just have to find the document using indexing and append the value)** |
| **Finding relation is by joining the relation table** | **Getting relation is faster in mongodb if the reference is stored inside the document, but if the reference is stored in another document mongodb has to find that document bu iterating this will be slow and sql will be fast as it just have to** |
| **Faster for more complex queries** | **Aggregations are faster in mongodb** |

Conclusion

|  |  |
| --- | --- |
| SQL | NOSQL |
| **Faster in** selecting records by non-key attributes | **Faster in** selecting with key (indexing) |
| Sorting | Inserting |
| Complex queries | updating |
|  | referencing |

Note: For below application we choose mongo bcoz of flexibity and better performance for our most common use cases







