MongoDB responded to these challenges by creating a technology foundation that enables development teams through:

1. The document data model – presenting them **the best way to work with data**.
2. A distributed systems design – allowing them to **intelligently put data where they want it**.
3. A unified experience that gives them the **freedom to run anywhere** – allowing them to future-proof their work and eliminate vendor lock-in.

With these capabilities, you can build an Intelligent Operational Data Platform, underpinned by MongoDB. In this Guide, we dive deeper into the architecture of MongoDB that enables the three technology foundations described above.

The Best Way to Work with Data: The Document Model

Relational, tabular databases have a long-standing position in most organizations. This made them the default way to think about storing, using, and enriching data. But enterprises are increasingly encountering limitations of this technology. Modern applications present new challenges that stretch the limits of what's possible with a relational database.

As organizations seek to build these modern applications, they find that the key differentiator for success is their development teams. Developers are on the front lines of digital transformation, and enabling them to work faster produces compounding benefits for the organization. To realize the full potential of data and software, developers turn to technologies that enable rather than hinder them.

Through strategies such as Agile and DevOps, microservices, cloud replatforming and more, many organizations have made significant progress in refactoring and evolving application tier code to respond faster to changing business requirements. But they then find themselves hampered by the rigidity and complexity of tabular databases.

Organizations need a fresh way to work with data. In order to handle the complex data of modern applications and simultaneously increase development velocity, the key is a platform that is:

- **Easy**, letting them work with data in a natural, intuitive way, supported by strong data consistency and transactional data guarantees
- **Flexible**, so that they can adapt and make changes quickly
- **Fast**, delivering great performance with less code
- **Versatile**, supporting a wide variety of data models, relationships, and queries

MongoDB's document model delivers these benefits for developers, making it the best way to work with data.

Easy: Natural, Intuitive, Transactional

Relational databases use a tabular data model, storing data across many tables. An application of any complexity easily requires hundreds or even thousands of tables. This sprawl occurs because of the way the tabular model treats data.

The conceptual model of application code typically resembles the real world. That is, objects in application code, including their associated data, usually correspond to real-world entities: customers or users, products, IoT connected assets, and so on. Relational databases, however, require a different structure. Because of the need to normalize data, a logical entity is typically represented in many separate parent-child tables linked by foreign keys. This data model doesn't resemble the entity in the real world, or how that entity is expressed as an object in application code.

This difference makes it difficult for developers to reason about the underlying data model while writing code, slowing down application development; this is sometimes referred to as *object-relational impedance mismatch*. One workaround for this is to employ an object-relational mapping layer (ORM). But this creates its own challenges, including managing the middleware and revising the mapping whenever either the application code or the database schema changes.