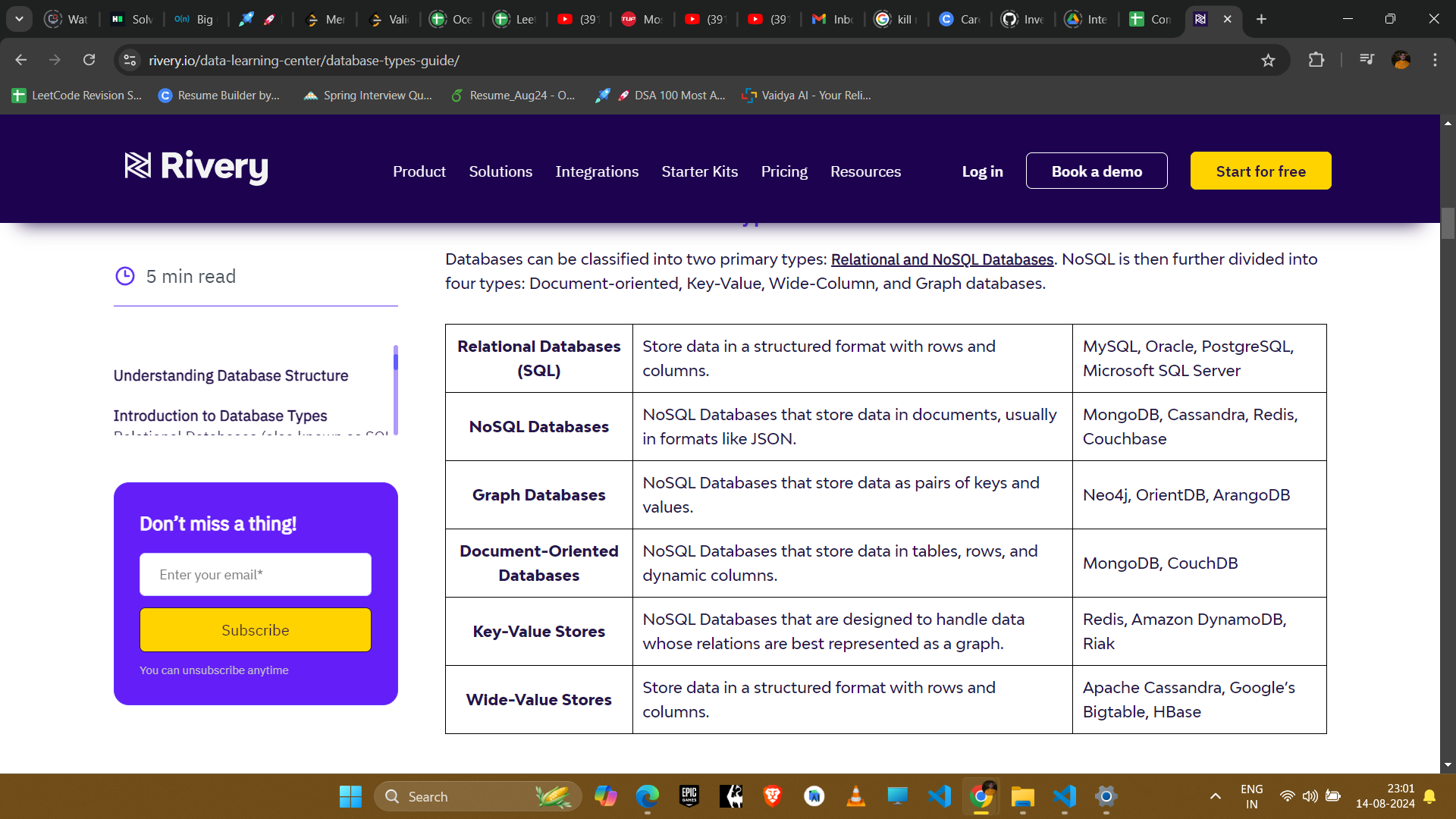
**Types of Database** (Source: <https://rivery.io/data-learning-center/database-types-guide/>)



**Relational Databases (also known as SQL Databases)**

A relational database (RDB) is a method of organizing data into tables, rows, and columns to show relationships between data points. This structure makes it straightforward to access, create, read, modify, and delete data using a querying language—such as SQL.

Each table, also known as a relation, has rows (records) and columns (fields), where each row represents an entity, and each column represents an attribute of that entity.

You can establish relationships between entities through primary and foreign keys by guaranteeing data integrity and enabling complex queries.

**Strengths of Relational Databases:**

* The structured, table-like schema is easy to understand.
* Follow ACID (Atomicity, Consistency, Isolation, Durability) properties which makes them reliable.
* SQL language is standardized, widely used, and applicable to a variety of database management systems.

**Weaknesses of Relational Database**s:

* Can be difficult to scale out on multiple servers (horizontal scaling).
* Each table requires a predefined schema which means all data inserted into the table must follow the same structure. This is not optimal when dealing with complex data structures.

**Common Use Cases:**

* Widely used in the financial industry thanks to the ACID properties that ensure data reliability in financial transactions and support complex financial analysis.
* In healthcare, relational databases are used to manage structured data such as patient records, medical histories, and test results, aiding healthcare workflows.
* Used to manage structured data such as customer, order, product, and payment data in e-commerce.

**NoSQL Databases**

NoSQL databases were developed as an alternative to traditional SQL databases, NoSQL databases are especially useful when working with large or fast-moving data that may not fit neatly into a table.

NoSQL databases use various data models for accessing and managing data. These databases are optimized for applications needing flexible data models, handling large volumes of data, and achieving low latency.

They accomplish this by relaxing some of the data consistency restrictions found in relational databases, making them ideal for dynamic, high-performance applications that require scalability and speed.

Instead of tables, NoSQL databases use more flexible data models, such as key-value pairs, documents, or graphs. They offer scalability and flexibility, making them suitable for handling large amounts of unstructured or semi-structured data. Examples include MongoDB, CouchBase, Cassandra, and Redis.

**Strengths of NoSQL Databases:**

* Flexible and scalable, ideal for handling large amounts of unstructured or semi-structured data.
* Scale horizontally across multiple servers, making them highly scalable and fault-tolerant.
* Well-suited for use cases such as social media, e-commerce, and big data analytics.

**Weaknesses of NoSQL Databases:**

* Lack of standardized query language: each type of NoSQL database has its own method for querying data, which can make these systems more challenging to learn, integrate, and communicate with.
* Not well-suited for complex transactions or querying relationships between data.

**Common Use Cases:**

* Social media platforms employ NoSQL databases to store and manage user-generated content, facilitating efficient storage and processing of diverse content types.
* In e-commerce, NoSQL databases are crucial for handling product recommendations and customer analytics, enabling personalized recommendations and improved customer experiences.
* NoSQL databases excel in big data analytics, handling high-velocity and high-volume data streams and extracting valuable insights from rapidly changing data sources.

**Relational vs NoSQL Databases**

SQL (Structured Query Language) and NoSQL (Not Only SQL) databases are fundamentally different in how they are designed, how they store data, and how they are used.

* **SQL databases** use a structured, tabular schema, adhere to ACID (Atomicity, Consistency, Isolation, and Durability) properties, and are optimized for complex queries and relationships, but they typically scale vertically, which can have limitations and higher costs.
* **NoSQL databases** offer flexibility with a schema-less structure and various data types, scale horizontally for handling large data volumes, and provide simpler query languages, but they may sacrifice ACID compliance for performance and are not uniformly designed for handling complex relationships.

