An Introduction to SQL Databases for Beginners

If you’re building an application that needs to store, organize and manage data, SQL databases are likely the best tool for the job. SQL (Structured Query Language) databases have been battle tested for decades and remain the go-to choice for relational data.

In this beginner’s introduction, we’ll answer common questions like:

- What are relational databases and SQL?
- Why are SQL databases so popular?
- When should you not use a SQL database?
- What are some key SQL concepts?

By the end, you’ll have a solid understanding of how SQL databases work and their many advantages. Let’s get started!

An Introduction to Relational Databases

Relational databases represent data in tabular format - rows and columns. Each row contains a data record with columns representing attributes of that record. For example, a “users” table might have columns like “name”, “email”, “location”, etc.

Relationships can be defined between tables. For instance, a “posts” table may have a “user_id” column that relates to the “id” column in the “users” table.

SQL (Structured Query Language) is the language used for defining tables, inserting data, querying, and managing relational databases. SQL offers a flexible, declarative way to manipulate dataset.

Why Use SQL Databases?

There are many good reasons SQL databases are the first choice for many applications:

- Flexible schema evolution - SQL makes it easy to modify table structures as requirements change.
- Powerful querying - Retrieve precise datasets with complex filters, aggregations, joins across tables.
- Reliable transactions - Atomicity, Consistency, Isolation, and Durability (ACID) protect data integrity.
- Scalability - Horizontally scale via partitioning and replicating across servers.
- Battle-tested - SQL databases have been around since the 1970s with constant optimization.

When Not to Use SQL

NoSQL databases like MongoDB have benefits for certain use cases:

- When data is highly unstructured or schema frequently changes.
- When simplicity of key-value data access is important.
- When hyper scale write performance is required.
- When sacrificing ACID transactional properties is acceptable.

For the majority of applications, SQL offers great flexibility. But evaluate NoSQL alternatives for the use cases above.

Key SQL Concepts

Here are some key concepts for getting started with SQL:

- Tables store data records (rows) with attributes (columns).
- Data types like VARCHAR, INTEGER, DATE, etc set column rules.
- Primary keys uniquely identify rows. Foreign keys relate tables.
- SELECT, INSERT, UPDATE, DELETE statements manipulate data.
- Joins, aggregation functions, views, stored procedures, etc extend capabilities.

While this guide just covers the basics, mastering these foundational SQL concepts will give you a great starting point for leveraging SQL databases effectively. The robust ecosystem of tools and decades of optimization make SQL a safe choice for most applications involving structured data.