# Classify your data

An online retail business has different types of data. Each type of data might benefit from a different storage solution.

Application data can be classified in one of three ways: structured, semi-structured, and unstructured. Here, you'll learn how to classify your data so that you can choose the appropriate storage solution for the type of data.

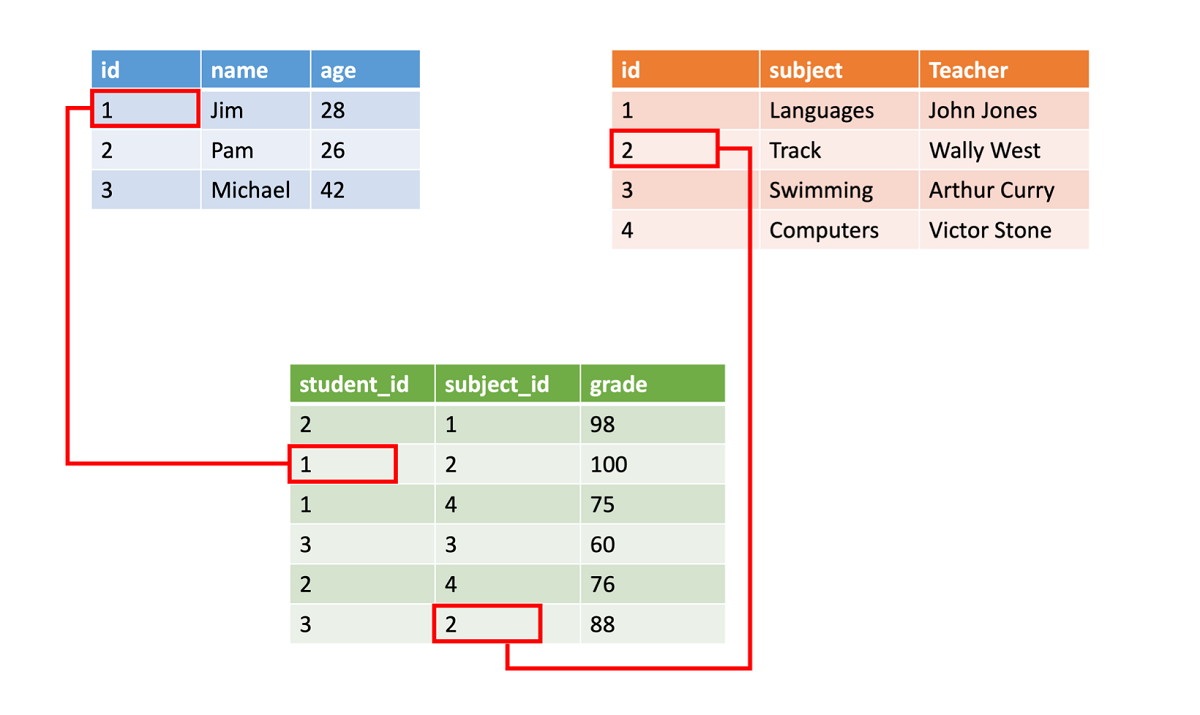
**Approaches to storing data in the cloud**

The following video introduces your options for storing data in the cloud:

**Structured data**

In structured data, sometimes called *relational data*, all data has the same fields or properties. All the data has the same organization and shape, or *schema*. The shared schema allows this type of data to be easily searched by using query languages like Structured Query Language (SQL). This capability makes this data style perfect for applications like CRM systems, reservations, and inventory management.

Structured data often is stored in database tables with rows and columns. In the table, a key column indicates how one row in a table relates to data in another row of another table. In the following image, a table that has data about grades gets data from a table of student names and a table of class data by using key columns.



Structured data is straightforward in that it's easy to enter, query, and analyze. All the data is in the same format. However, forcing a consistent structure also means that the evolution of the data is more difficult. If you add or remove data fields, you must update each record to conform to the new structure.

**Semi-structured data**

Semi-structured data is less organized than structured data. Semi-structured data isn't stored in a relational format because the fields don't fit neatly into tables, rows, and columns. Semi-structured data contains tags that make the organization and hierarchy of the data apparent. One example is key/value pairs. Semi-structured data is also referred to as non-relational or *not only SQL (NoSQL)* data.

Semi-structured data is defined by a data serialization language. In data classification, *serialization* is the process of converting data into a format that can be transmitted or stored.

Software developers use data serialization languages to write data stored in memory to a file, which can then be sent to another system, parsed, and read. The sender and receiver don’t need to know details about the other system. As long as the same serialization language is used, the data can be understood by both systems.

**Common serialization languages**

Three common serialization languages are XML, JSON, and YAML.

**XML**

*Extensible Markup Language (XML)* was one of the first data languages to be widely used. XML is text-based, which makes it easily human-readable and machine-readable. XML parsers are available for almost all popular development platforms.

You can use XML to express relationships. XML has standards for schema, transformation, and even displaying on the web.

Here's an example of a person's name, age, and hobbies expressed in XML:

XMLCopy

<Person Age="23">

<FirstName>Quinn</FirstName>

<LastName>Anderson</LastName>

<Hobbies>

<Hobby Type="Sports">Golf</Hobby>

<Hobby Type="Leisure">Reading</Hobby>

<Hobby Type="Leisure">Guitar</Hobby>

</Hobbies>

</Person>

XML expresses the shape of the data by using *tags* that are defined inside angle braces. The tags come in two forms: *elements* such as <FirstName> and *attributes* that can be expressed in text like Age="23". Elements can have child elements to express relationships. For example, the <Hobbies> tag expresses a collection of Hobby elements.

XML is flexible and can express complex data easily. However, it tends to be more verbose, which makes it larger to store, process, and pass over a network. As a result, other formats have become more popular.

**JSON**

*JavaScript Object Notation (JSON)* has a lightweight specification and uses curly braces to indicate data structure. Compared to XML, JSON is less verbose, and it's easier for humans to read. JSON frequently is used by web services to return data.

Here's the same person's name, age, and hobbies expressed in JSON:

JSONCopy

{

"firstName": "Quinn",

"lastName": "Anderson",

"age": "23",

"hobbies": [

{ "type": "Sports", "value": "Golf" },

{ "type": "Leisure", "value": "Reading" },

{ "type": "Leisure", "value": "Guitar" }

]

}

The JSON format isn't as formal as XML. It's closer to a key/value pair model than to a formal data expression. As you might guess from the name, the JavaScript programming language has built-in support for this format, so it's popular for web development. Like XML, other languages have parsers you can use to work with this data format. The downside of JSON is that it tends to be more programmer-oriented, so it's harder for non-technical people to read and modify.

**YAML**

*YAML Ain’t Markup Language (YAML)* is a more recently developed data serialization language. One of the benefits of using YAML is that it's easier for humans to read than some other languages. The data structure is defined by line separation and indentation. The YAML format reduces the dependency on structural characters like parentheses, commas, and brackets.

Here's the same data expressed in YAML:

YAMLCopy

firstName: Quinn

lastName: Anderson

age: 23

hobbies:

- type: Sports

value: Golf

- type: Leisure

value: Reading

- type: Leisure

value: Guitar

This format is more readable than JSON, and it's often used for configuration files that need to be written by people but parsed by programs. YAML is the newest of these data formats.

**What is semi-structured or NoSQL data?**

The following video describes semi-structured data and NoSQL data storage options:

**Unstructured data**

The organization of unstructured data is undefined. Unstructured data is often delivered in file format, such as in photo or video files. The video file itself might have an overall structure and come with semi-structured metadata, but the data that forms the video itself is unstructured. Therefore, photos, videos, and other similar files are classified as unstructured data.

Examples of unstructured data include:

* Media files, like photos, videos, and audio files
* Microsoft 365 files, like Word documents
* Text files
* Log files

**Data classification: Evaluate your data types**

You can classify data in one of three ways: structured, semi-structured, and unstructured. Understanding the differences so that you can classify your data helps you choose the correct storage solution.

Structured data is organized data that neatly fits into tables or columns of data. Semi-structured data is still organized and has clear properties and values, but there's variety to the data. Unstructured data doesn't fit neatly into tables or columns, and it doesn't have a uniform schema.

Let's look at the datasets used in an online retail business and classify them.

**Product catalog data**

Product catalog data for an online retail business is semi-structured in nature. Each product has a product SKU, a description, a quantity, a price, size options, color options, a photo, and possibly a video. So, this data appears relational to begin with because it all has the same structure. However, as you introduce new products or different kinds of products, you might want to add data fields. For example, new tennis shoes you carry are Bluetooth-enabled to relay sensor data from the shoe to a fitness app on the user’s phone. This feature appears to be a growing trend, and you want to give customers the option to filter on "Bluetooth-enabled" shoes. You don't want to update all your existing shoe data with a Bluetooth-enabled property. You want to add this new property only to new shoes.

With the addition of the Bluetooth-enabled property, your shoe data is no longer homogenous. You've introduced differences in the schema. If this change is the only exception you expect to encounter, you can normalize the existing data so that all products included a "Bluetooth-enabled" field to maintain a structured, relational organization. However, if it's just one of many specialty fields that you envision supporting in the future, the classification of the data is semi-structured. The data is organized by tags, but each product in the catalog can contain unique fields.

The classification for product catalog data is *semi-structured*.

**Photos and videos**

The photos and videos displayed on product pages are unstructured data. Although the media file might contain metadata, the body of the media file is unstructured.

The data classification for photos and videos is *unstructured*.

**Business data**

Business analysts want to implement business intelligence to perform inventory pipeline evaluations and sales data reviews. To perform these operations, data from multiple months needs to be aggregated and then queried. Because of the need to aggregate similar data, this data must be structured, so that one month can be compared to the next.

The classification for business data is *structured*.