**Meaning of Data Alignment Intrinsic**

Data alignment being “intrinsic” means that **it is an inherent property of the operations performed on data, rather than something that can be explicitly controlled or manipulated by the programmer**. In other words, data alignment is a natural consequence of how data is stored and accessed in memory, driven by the underlying architecture and hardware.

In computer systems, data alignment refers to the process of organizing data in memory to optimize access and retrieval. This involves ensuring that data elements, such as integers or floating-point numbers, are stored at addresses that are multiples of their respective word sizes (e.g., 4-byte alignment for 32-bit integers).

When data alignment is intrinsic, it means that **the programming language, compiler, or runtime environment automatically handles data alignment for you, without requiring explicit instructions or directives from the programmer**. This is often the case with modern programming languages and libraries, which are designed to take advantage of the underlying hardware architecture and memory management systems.

For example, in languages like C and C++, the compiler will typically align data structures and variables according to the target platform’s memory alignment requirements, without requiring the programmer to specify explicit alignment directives.

Intrinsic data alignment has several benefits, including:

1. **Improved performance**: By aligning data naturally, the CPU can access and process data more efficiently, reducing memory access latency and increasing overall system performance.
2. **Simplified programming**: Programmers don’t need to worry about explicit data alignment, allowing them to focus on higher-level logic and algorithmic design.
3. **Portability**: Intrinsic data alignment ensures that programs written for one platform can be easily ported to another, without requiring modifications to handle different memory alignment schemes.

In summary, “data alignment is intrinsic” means that data alignment is an automatic process, driven by the underlying hardware and software architecture, rather than something that can be explicitly controlled by the programmer. This simplifies programming, improves performance, and enhances portability across different platforms.

Pandas support duplicate values and also support duplicate indexes. But it very stupid to choose or make that type of data. So either remove duplicates or just edit the values.

Normal table:

1. Not flexible
2. Shows entire data

Pivot table:

1. Flexible
2. Show summarized data.
3. Give insights.

**Data Preparation:**

Before creating a pivot table, it’s essential to ensure that the data is properly prepared and structured. Here are the key prerequisites to follow:

1. **Ensure a Consistent Data Structure**:
   * Your data should be organized in a tabular format, with **column headers** in the first row and each subsequent row representing a record.
   * Each column should have a unique, descriptive header (e.g., Company, Product, RAM (GB)), with no blank column headers.
2. **Remove Empty Rows and Columns**:
   * Ensure there are no completely empty rows or columns within your dataset, as these can disrupt the pivot table generation process.
3. **Standardize Data Types**:
   * Check that each column has a **consistent data type**:
     + Numeric data (e.g., Price (Euro), Inches, RAM (GB)) should contain only numbers.
     + Text columns (e.g., Company, Product, OpSys) should be standardized to avoid inconsistencies like extra spaces or case variations (e.g., "Dell" vs. "dell").
   * **Convert dates** to an Excel-recognized date format if applicable.
4. **Handle Missing Values**:
   * **Identify and address missing values** in your data, as they may cause issues or misinterpretations.
   * You can either fill missing values with suitable substitutes (e.g., using 0 for missing numerical values or N/A for text fields) or remove incomplete rows if they are minimal and unimpactful.
5. **Remove Duplicates**:
   * Check for and remove **duplicate rows** that may skew results, especially if duplicate records aren’t intentional or useful for your analysis.
6. **Format Data as a Table (Optional)**:
   * Converting your data range to an **Excel Table** (via **Home > Format as Table**) can make managing the pivot table easier, as any updates to the table will automatically reflect in the pivot table.
7. **Name Your Table Range (Optional)**:
   * Assign a name to your data range (e.g., ProductData) under **Formulas > Define Name**. This helps keep your dataset organized and makes it easier to reference in multiple pivot tables.

**Basic Definition:**

Values = Means # what to analyse?

Aggfunc = Means # How to analyse?

Index = Represents the # Rows.

In the context of Excel:

‘Values’ section contains the value that we need to pass to either make sum of ‘field’ or ‘Average’ of ‘field’.

Example: How does price vary between different operating systems?

Meaning: I need to find the price variation for each operating system. So I have to find average price first. So in value section price column will be selected.

Then in row column the main field ‘which data is giving us insight. Here the main differentiator is ‘os’.