

## Lecture Notes

# How Data Exists Within Enterprises

To meaningfully analyse the wide range of data that any organisation has, it needs to be in a format so that it can be analysed easily. Data warehousing is the study of systematically organising the way we store data.

### Defining Data Warehouse

A data warehouse is a collection of data with the following properties:

1. **Subject-oriented:** Data is categorised and stored by business subject rather than by the source, i.e. the data in the data warehouse should pertain to the particular problem you are trying to solve.
2. **Integrated:** Data is stored in various platforms in an organisation and you need to integrate/bring the data residing in various databases to a unified platform which can help you solve the business problem. It is constructed by integrating multiple heterogeneous data sources.
3. **Non-volatile:** Typically, data in the data warehouse is not deleted or altered without a legitimate reason.
4. **Time variant:** Data is updated with the variation in time.

A data warehouse is essential for an enterprise. It gives an overall view of the entire organisation and makes it easy to carry out analytical operations on the data.

### OLAP vs OLTP

A transactional database is needed for performing day-to-day activities. A data warehouse is also a database, although it is very different from a transactional database in many ways. Some of the differences are tabulated in figure 1.

## Difference between Transactional Database and Data Warehouse

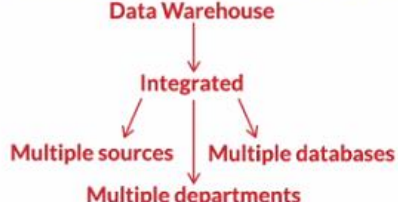
	Transactional Database	Data Warehouse
<b>Purpose</b>	1. To perform business transactions	1. To conduct analysis and decision making
<b>User</b>	1. End users - for business transactions	1. Managers, CEOs, CXOs - for decision making
<b>Technology</b>	1. No ETL process 2. DBMS 3. Service side application - API	1. ETL process - Extract, Transform and Load process 
<b>Front-end Technology</b>	1. Web servers 2. Browsers for interaction 3. Mobile phones 4. Interface with payment gateways and payment systems 5. No BI a. Canned reports	1. BI tools: Business Intelligence a. Cognos b. Tableaut c. Reporting front-end infrastructure
<b>Database Design</b>	1. Based on integrating issues: Normalisation 2. No star schema 3. Nature of Access : CRUD - Create, Retrieve, Update, Delete	1. Based on dimensional modelling 2. Creation of star schema 3. Nature of Access : Retrieve
<b>Also known as</b>	1. OLTP: Online Transaction Processing systems	1. OLAP: Online Analytical Processing systems

Figure 1 - Difference between Transactional Database and Data Warehouse Database

### Structure of Data Warehouse

You have learnt about what a data warehouse is. One of the primary methods of designing a data warehouse is called **dimensional modelling**. The two key elements of dimensional modelling are **facts** and **dimensions**. Numeric attributes are facts while dimension variables are non-numeric variables and act a meta data corresponding to fact variables.

A typical problem might involve multiple databases with many different variables, while you may not be interested in all the variables. Hence, only some facts and dimensions are combined in a specific manner to create the structure of data warehouse. This structure is called as a schema diagram.

### Star Schema

Star schema is a structure which can be used to put facts and dimensions together. The Star schema structures the data in a way that it deals only with dimensions that are *directly* connected to the fact they explain. Basically, there is only one layer of metadata attached to the fact variables. They are named star schema because they take the structure of a star.

The central table containing the basic sales information is the central fact table. It has dimension variables such customer name, product name, etc. To understand these dimension variables better, what you need

are dimension tables specific to the dimension variable. Figure 2 explains the structure of star schema using the example of e-commerce sales.

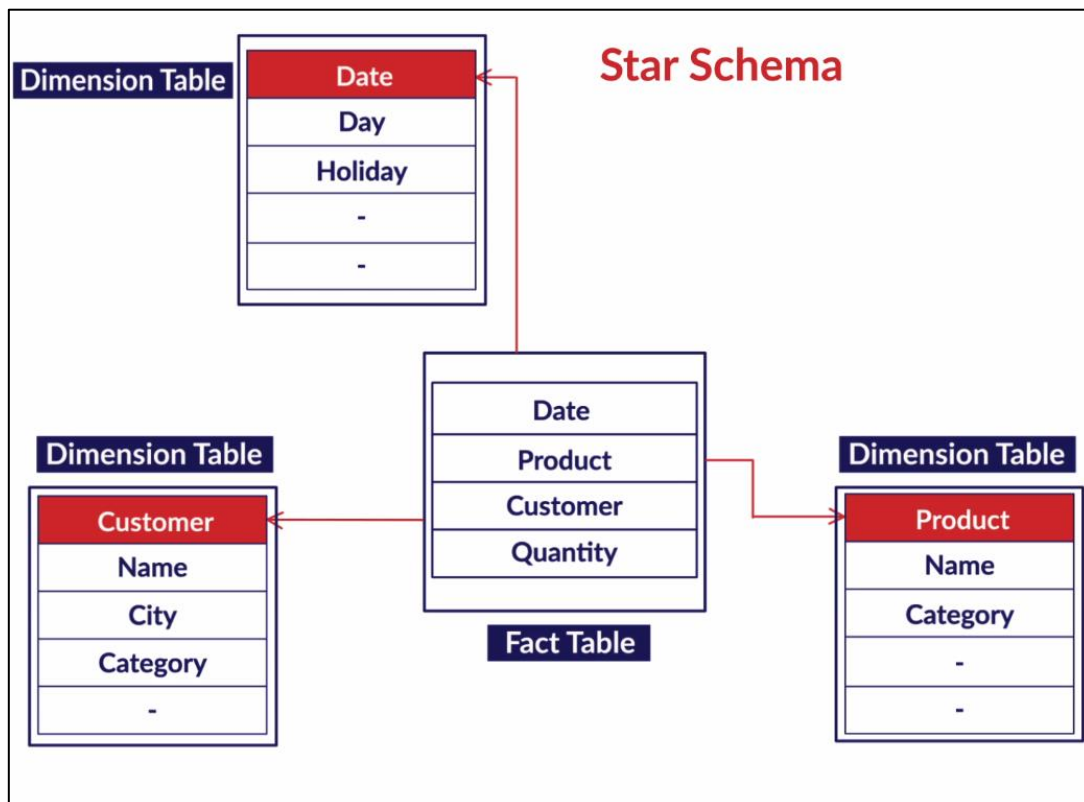


Figure 2 - Structure of Star Schema

## ETL operations

ETL – Extract, Transform, Load are processes used for extracting data from multiple different databases.

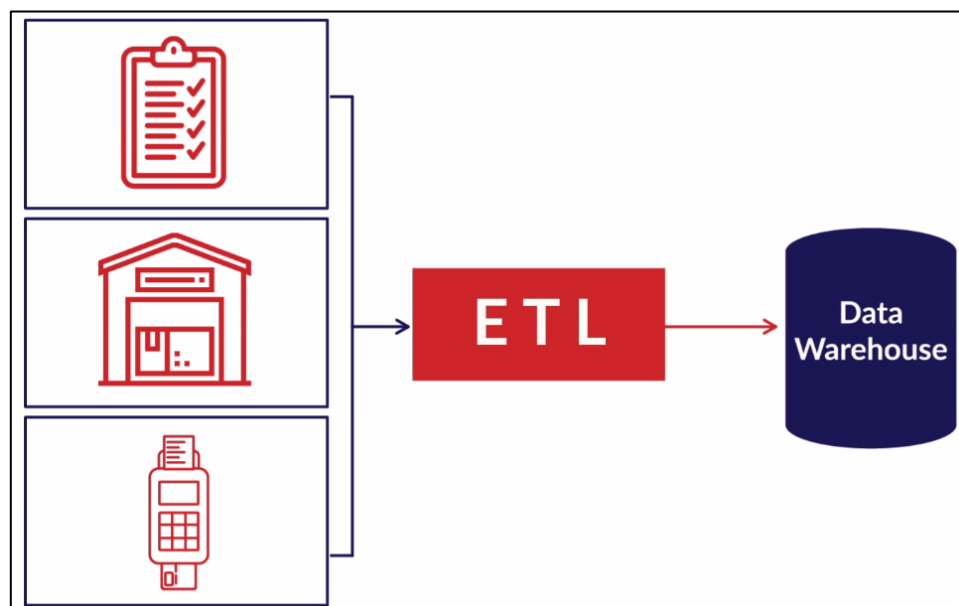


Figure 3 – Purpose of ETL operations

Extracted data from multiple sources must be transformed first to the format we want in the data warehouse, since it is coming from different sources and there might be differences in the way data is interpreted in the original sources. Finally, it is loaded in the data warehouse structure.

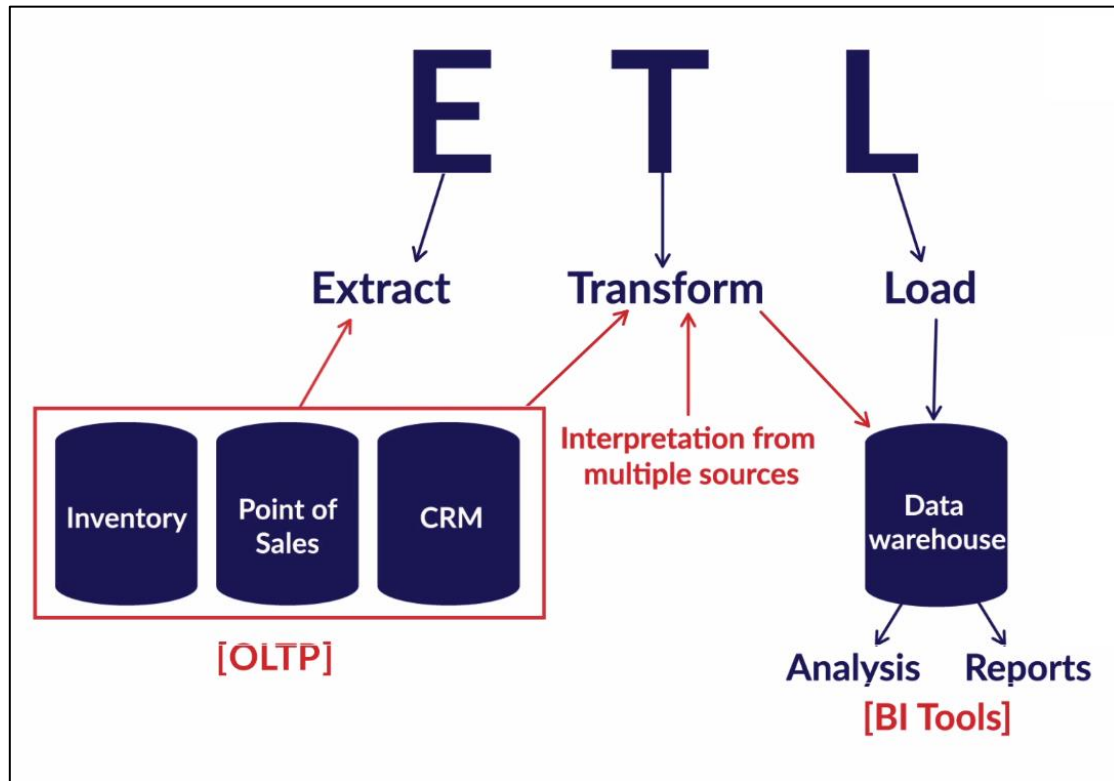


Figure 4 – Different components of ETL operations

## Business Intelligence

Business Intelligence is nothing but using tools such as Tableau, Pentaho, etc. to connect to the data warehouse back end and performing visual analytics and making reports. Figure 5 explains what BI is.

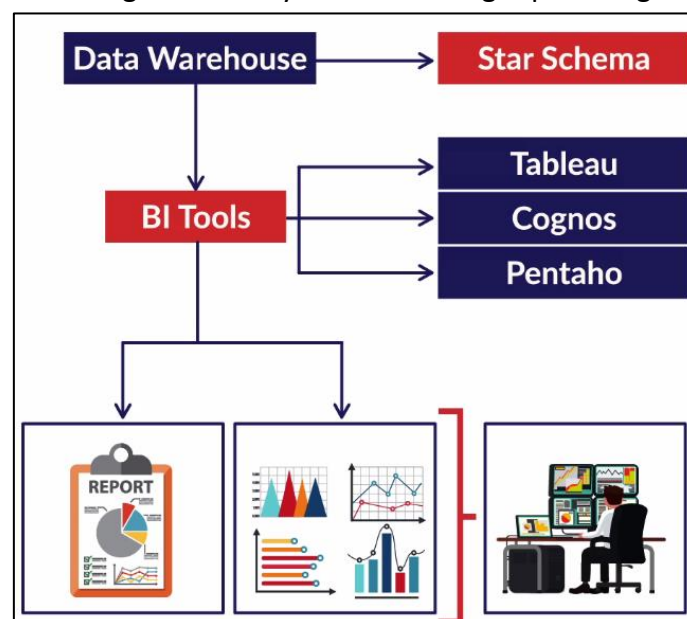


Figure 5 – Business Intelligence

These reports and analysis are used by the management for taking decisions, that is why these are known as Decision Support Systems.

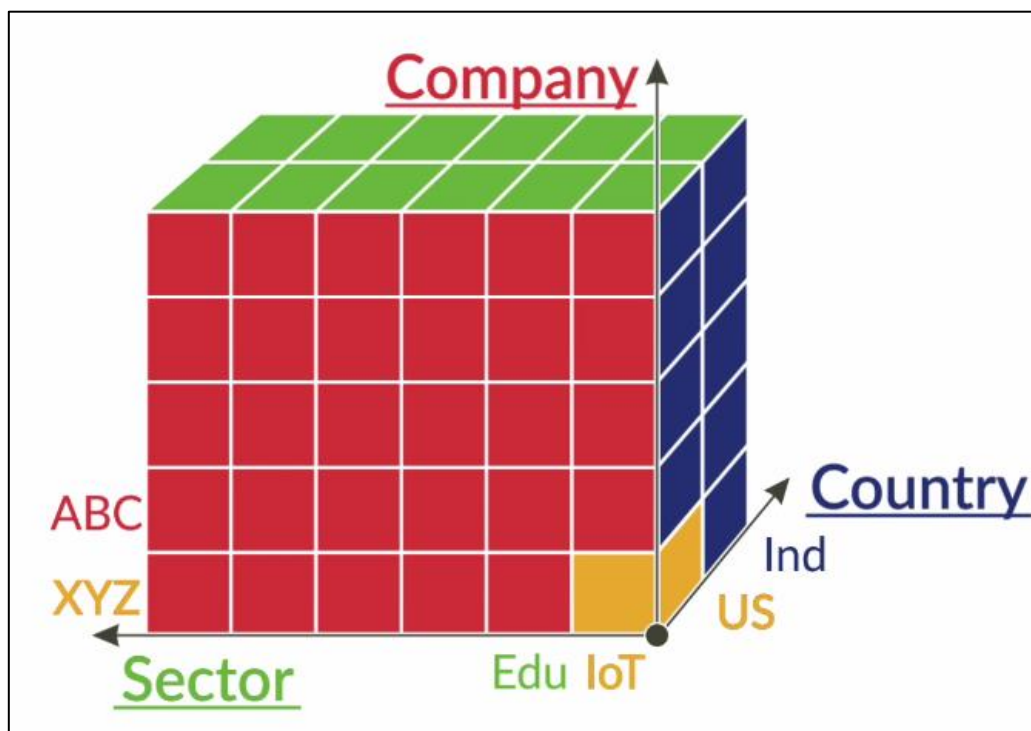
## Data Cubes

Schemas are structures to store data in a systematic fashion. When data is stored into these schemas, they become what we call as data cubes. Data cubes are used to store relevant information in such a way that they can be used to summarise data per the needs.

- The dimension variables are visualised on the axes and fact data pertaining to those dimensions is stored inside the cube itself.
- A data cube can only be visualised for three dimensions but ideally it can have any number of dimensions.

As you can see in the infographic below, the dimensions of company name, country name and sector are represented on the three axes, and the fact variable, which is the investment amount related to these dimensions, is stored inside the cells of the cube. Here, the fact data pertaining to company XYZ, country US and sector IOT is highlighted in the image below.

Figure 6 shows data from the startup investments example visualized inside a data cube.



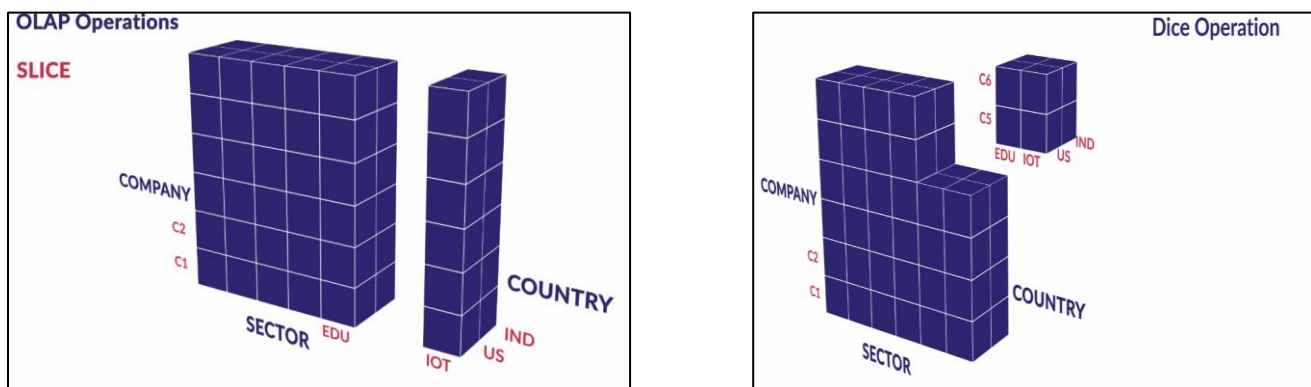
**Figure 6 - Data Cube with 3 Dimensions: Company, Sector and Country and Investment Amount in Data Cells**

## OLAP Operations

Data can be organised in the form of data cubes so that it can be retrieved quickly. Hence, it becomes important to understand the forms in which data can be retrieved and the operations that you can perform to extract information. It's important to know how to extract relevant data because to solve any business problem, you should first understand the entire data in the data warehouse and access parts of it to do so. Operations such as slice, dice, roll up and drill down are used for exactly this purpose.

The following operations are used to subset to obtain relevant information.

- Slice: Data is filtered based on one specific dimension
- Dice: Data is filtered on more than one dimension



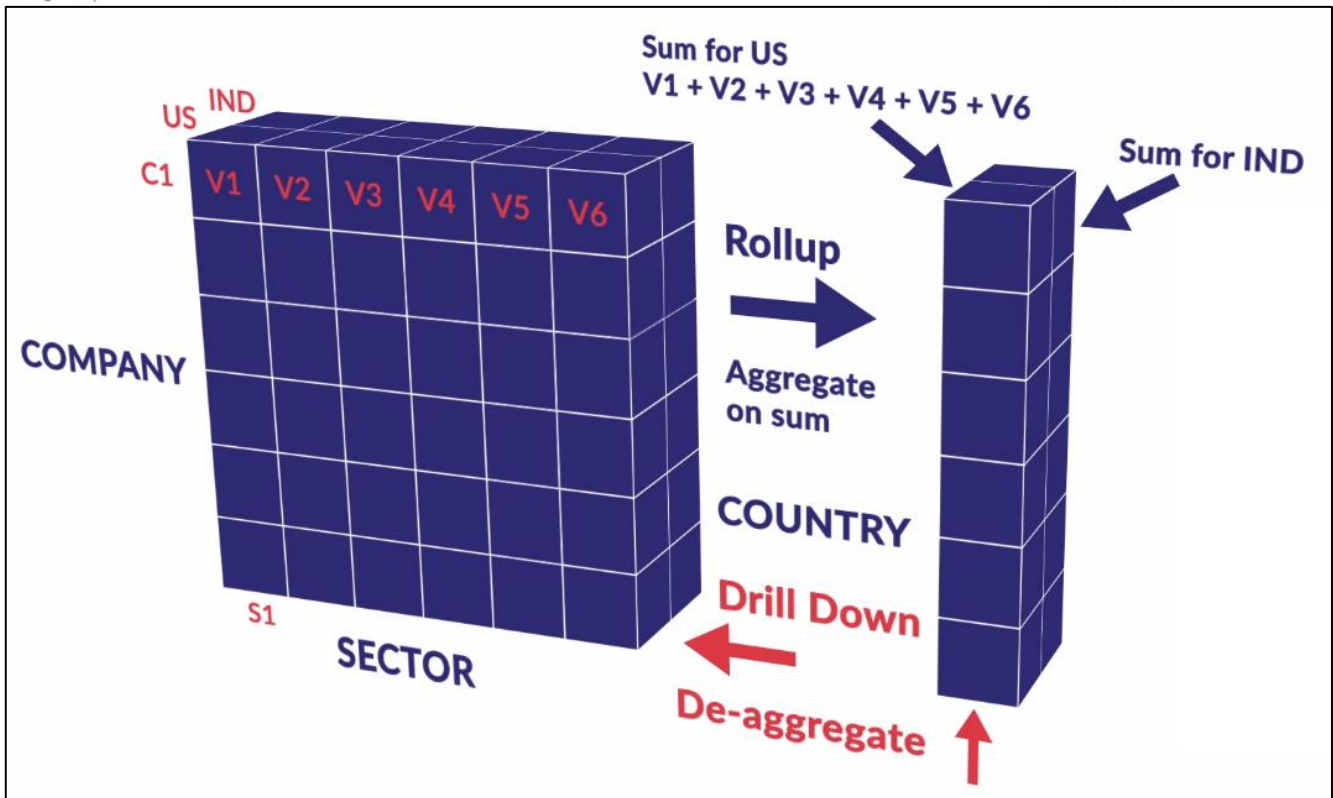
**Figure 7 - Visual Representation of Slice and Dice Operations**

When slice and dice operations are used, the data is extracted in the original form in which it is stored in the data warehouse. Basically, no aggregate operations are performed on the data. In slicing and dicing, you retrieve information from some specific cells within the data cube. Let's recall what roll up and drill down is:

- Roll up: Data is summarised on one or more dimensions
- Drill down: Summarised data is de-aggregated on one or more dimensions

When you perform a roll up operation, aggregations such as sum or average are performed on the data based on one or more dimensions. Drill down is the opposite of roll up, i.e. de-aggregation. Refer to figure 8.





**Figure 8 - Visual Representation of Rollup and Drill Down Operations**

The terminologies for the aggregation operations on a data cube are slightly different. They are:

- Edge of the cube: When aggregation is done on all dimensions except one dimension, an edge is obtained as the result of the aggregation operation. It is basically obtained when roll up is done on a data cube based on all but one dimensions.
- Face of the cube: When aggregation is done on all dimensions except two dimensions, a face is obtained as the result of the aggregation operation.

Dimension variables serve as the entry point for analysis.

### Superstore Sales Case & Lab

In this session, you applied your understanding of data warehousing, dimension modelling and OLAP operations on a real-world data set. You got some hands-on practice of the various OLAP operations in both Excel and R. You learnt how to concatenate facts and dimensions using the VLOOKUP function in Excel and the merge command in R. You then learnt to use the subset and aggregate functions to perform slice, dice and roll up in R. Similar things were done in Excel using the filter function and the pivot tables and the merged files were transferred from R to Excel using the write.table command.

Merging in Excel is done using the VLOOKUP function; it is used to look up a value in a table by matching on the first column. It returns the matched value from a table. The syntax is as follows:

**VLOOKUP (value, table, col\_index, [range\_lookup])**

- value: The value to look for in the first column of a table.
- table: The table from which to retrieve a value.

- `col_index`: The column in the table from which to retrieve a value.
- `range_lookup`: [optional] TRUE = approximate match (default). FALSE = exact match.

This will merge only one column which will be the value argument provided in the VLOOKUP function. To merge multiple columns, this method is inefficient as the function must be used multiple times, thus R is used.

Slice and dice are achieved using the FILTER function in Excel.

Merging in R is done using the merge command. It merges two data frames by common column or row names. The function looks as follows:

`merge(x, y, ...)`

- `x` and `y` are the data frames to be merged

Following this are the other arguments which help us obtain the inner, outer, left or right merge as required. The arguments are `all.x`, `all.y` and `all`. The default for all these arguments is FALSE which gives us the inner merge. To obtain any other type of merge the following arguments are changed:

Argument	Type of Merge
<code>all.x = TRUE</code>	Left Merge
<code>all.y=TRUE</code>	Right Merge
<code>all=TRUE</code>	Outer Merge

Slice and dice are achieved using the subset command in R. The function looks as follows when we want to subset on a data frame:

`subset(x, subset, select, drop = FALSE, ...)`

- `x` is the object to be subsetted.
- `subset`: logical expression indicating elements or rows to keep: missing values are taken as false.
- `select`: expression, indicating columns to select from a data frame
- `drop`: passed on to the indexing operator.
- `...`: further arguments to be passed to or from other methods.

Roll ups are achieved using aggregate function in R. The function looks as follows:

`aggregate(x, by, FUN, ..., simplify = TRUE, drop = TRUE)`

- `x` is an R object
- `by`: a list of grouping elements, each as long as the variables in the data frame `x`. The elements are coerced to factors before use.
- `FUN`: a function to compute the summary statistics which can be applied to all data subsets.
- `simplify`: a logical indicating whether results should be simplified to a vector or matrix if possible. Default is TRUE.



- **drop**: a logical indicating whether to drop unused combinations of grouping values. Default is TRUE.

It is important to understand what the output of the merge will contain per the specifications of left, right, inner or outer merge. Sometimes discrepancies might appear. For example, the number of observations might not be equal in the merged and the original file after performing inner merge.

Files can be exported from R to the disk using the `write.table` function. The `write.table` command is used when you want the output file in a text format, `write.csv` command can be used to write a file in csv format. The function looks as follows:

`write.table(x, file = "", append = FALSE, quote = TRUE, sep = " ", na = "NA", row.names = TRUE, col.names = TRUE, ...)`

- **x** is the object to be written, preferably a matrix or data frame. If not, it is attempted to coerce x to a data frame.
- **file**: either a character string naming a file or a connection open for writing. "" indicates output for the console.
- **append**: logical. Only relevant if file is a character string. If TRUE, the output is appended to the file. If FALSE, any existing file of the name is destroyed.
- **quote**: a logical value or a numeric vector. If TRUE, any character or factor columns will be surrounded by double quotes. If a numeric vector, its elements are taken as the indices of columns to quote. In both cases, row and column names are quoted if written. If FALSE, nothing is quoted.
- **sep** : the field separator string. Values within each rows of x are separated by this string.
- **na** : the string to use for missing values in the data
- **row.names** : either a logical value indicating whether the row names of x are to written along with x, or a character vector of row names is to be written.
- **col.names** : either a logical value indicating whether the column names of x are to written along with x, or a character vector of column names is to be written.