Data Frame

DataFrame is a two-dimensional labeled data structure with columns of which can be of different types.

DataFrame is the most important data stucture in the field Data analytics and Data Science.

Data can be collected from various source like Database, Cloud, CSV, etc, but is ultimately stored in a dataframe.

Creating data frames from scratch

Data frame can be created using **data.frame()** function

In [1]:
```
v1 <- 1:5
```

In [1]:
```
# Letter is an inbuilt list
letters
```

'a' 'b' 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' 'k' 'l' 'm' 'n' 'o' 'p' 'q' 'r' 's' 't' 'u' 'v' 'w' 'x' 'y' 'z'

In [3]:
```
v2 <- letters[1:5]
```

In [4]:
```
df <- data.frame(v1,v2)
```

In [5]:
```
df
```

v1	v2
1	a
2	b
3	c
4	d
5	e

Creating dataframe with column name

In [6]:
```
c1 <- c('Tendulkar','Kohli','Dohni','Bumrah','Chahal')
c2 <- c(10000,7100,5800,890,870)
c3 <- c(11,0,0,370,420)
```

In [7]:
```
cricket <- data.frame(players=c1,runs=c2,wickets=c3)
```

In [8]:
```
cricket
```

players	runs	wickets
Tendulkar	10000	11
Kohli	7100	0
Dohni	5800	0
Bumrah	890	370
Chahal	870	420

Inbuilt data frames

View inbuilt data frames using **data()** function

In [9]:
```
data()
```

In [10]:
```
state.x77
```

	Population	Income	Illiteracy	Life Exp	Murder	HS Grad	Frost	Area
Alabama	3615	3624	2.1	69.05	15.1	41.3	20	50708
Alaska	365	6315	1.5	69.31	11.3	66.7	152	566432
Arizona	2212	4530	1.8	70.55	7.8	58.1	15	113417
Arkansas	2110	3378	1.9	70.66	10.1	39.9	65	51945
California	21198	5114	1.1	71.71	10.3	62.6	20	156361
Colorado	2541	4884	0.7	72.06	6.8	63.9	166	103766
Connecticut	3100	5348	1.1	72.48	3.1	56.0	139	4862
Delaware	579	4809	0.9	70.06	6.2	54.6	103	1982
Florida	8277	4815	1.3	70.66	10.7	52.6	11	54090
Georgia	4931	4091	2.0	68.54	13.9	40.6	60	58073
Hawaii	868	4963	1.9	73.60	6.2	61.9	0	6425
Idaho	813	4119	0.6	71.87	5.3	59.5	126	82677
Illinois	11197	5107	0.9	70.14	10.3	52.6	127	55748
Indiana	5313	4458	0.7	70.88	7.1	52.9	122	36097
Iowa	2861	4628	0.5	72.56	2.3	59.0	140	55941
Kansas	2280	4669	0.6	72.58	4.5	59.9	114	81787
Kentucky	3387	3712	1.6	70.10	10.6	38.5	95	39650
Louisiana	3806	3545	2.8	68.76	13.2	42.2	12	44930
Maine	1058	3694	0.7	70.39	2.7	54.7	161	30920
Maryland	4122	5299	0.9	70.22	8.5	52.3	101	9891
Massachusetts	5814	4755	1.1	71.83	3.3	58.5	103	7826
Michigan	9111	4751	0.9	70.63	11.1	52.8	125	56817
Minnesota	3921	4675	0.6	72.96	2.3	57.6	160	79289
Mississippi	2341	3098	2.4	68.09	12.5	41.0	50	47296
Missouri	4767	4254	0.8	70.69	9.3	48.8	108	68995
Montana	746	4347	0.6	70.56	5.0	59.2	155	145587
Nebraska	1544	4508	0.6	72.60	2.9	59.3	139	76483
Nevada	590	5149	0.5	69.03	11.5	65.2	188	109889
New Hampshire	812	4281	0.7	71.23	3.3	57.6	174	9027
New Jersey	7333	5237	1.1	70.93	5.2	52.5	115	7521
New Mexico	1144	3601	2.2	70.32	9.7	55.2	120	121412
New York	18076	4903	1.4	70.55	10.9	52.7	82	47831
North Carolina	5441	3875	1.8	69.21	11.1	38.5	80	48798
North Dakota	637	5087	0.8	72.78	1.4	50.3	186	69273

	Population	Income	Illiteracy	Life Exp	Murder	HS Grad	Frost	Area
Ohio	10735	4561	0.8	70.82	7.4	53.2	124	40975
Oklahoma	2715	3983	1.1	71.42	6.4	51.6	82	68782
Oregon	2284	4660	0.6	72.13	4.2	60.0	44	96184
Pennsylvania	11860	4449	1.0	70.43	6.1	50.2	126	44966
Rhode Island	931	4558	1.3	71.90	2.4	46.4	127	1049
South Carolina	2816	3635	2.3	67.96	11.6	37.8	65	30225
South Dakota	681	4167	0.5	72.08	1.7	53.3	172	75955
Tennessee	4173	3821	1.7	70.11	11.0	41.8	70	41328
Texas	12237	4188	2.2	70.90	12.2	47.4	35	262134
Utah	1203	4022	0.6	72.90	4.5	67.3	137	82096
Vermont	472	3907	0.6	71.64	5.5	57.1	168	9267
Virginia	4981	4701	1.4	70.08	9.5	47.8	85	39780
Washington	3559	4864	0.6	71.72	4.3	63.5	32	66570
West Virginia	1799	3617	1.4	69.48	6.7	41.6	100	24070
Wisconsin	4589	4468	0.7	72.48	3.0	54.5	149	54464
Wyoming	376	4566	0.6	70.29	6.9	62.9	173	97203

Understanding data

Use **head() & tail()** function to see first 6 and last 6 rows respectively

In [11]:
```
head(state.x77)
```

	Population	Income	Illiteracy	Life Exp	Murder	HS Grad	Frost	Area
Alabama	3615	3624	2.1	69.05	15.1	41.3	20	50708
Alaska	365	6315	1.5	69.31	11.3	66.7	152	566432
Arizona	2212	4530	1.8	70.55	7.8	58.1	15	113417
Arkansas	2110	3378	1.9	70.66	10.1	39.9	65	51945
California	21198	5114	1.1	71.71	10.3	62.6	20	156361
Colorado	2541	4884	0.7	72.06	6.8	63.9	166	103766

In [12]:
```
tail(state.x77)
```

	Population	Income	Illiteracy	Life Exp	Murder	HS Grad	Frost	Area
Vermont	472	3907	0.6	71.64	5.5	57.1	168	9267
Virginia	4981	4701	1.4	70.08	9.5	47.8	85	39780
Washington	3559	4864	0.6	71.72	4.3	63.5	32	66570
West Virginia	1799	3617	1.4	69.48	6.7	41.6	100	24070
Wisconsin	4589	4468	0.7	72.48	3.0	54.5	149	54464
Wyoming	376	4566	0.6	70.29	6.9	62.9	173	97203

Use **summary()** and **str()** function to get basic description of data like mean,median,quartiles etc

In [13]:
```
summary(state.x77)
```

```
   Population        Income       Illiteracy       Life Exp    
 Min.   :  365   Min.   :3098   Min.   :0.500   Min.   :67.96  
 1st Qu.: 1080   1st Qu.:3993   1st Qu.:0.625   1st Qu.:70.12  
 Median : 2838   Median :4519   Median :0.950   Median :70.67  
 Mean   : 4246   Mean   :4436   Mean   :1.170   Mean   :70.88  
 3rd Qu.: 4968   3rd Qu.:4814   3rd Qu.:1.575   3rd Qu.:71.89  
 Max.   :21198   Max.   :6315   Max.   :2.800   Max.   :73.60  
     Murder          HS Grad          Frost             Area     
 Min.   : 1.400   Min.   :37.80   Min.   :  0.00   Min.   :  1049  
 1st Qu.: 4.350   1st Qu.:48.05   1st Qu.: 66.25   1st Qu.: 36985  
 Median : 6.850   Median :53.25   Median :114.50   Median : 54277  
 Mean   : 7.378   Mean   :53.11   Mean   :104.46   Mean   : 70736  
 3rd Qu.:10.675   3rd Qu.:59.15   3rd Qu.:139.75   3rd Qu.: 81163  
 Max.   :15.100   Max.   :67.30   Max.   :188.00   Max.   :566432  
```

In [14]:
```
str(state.x77)
```

```
 num [1:50, 1:8] 3615 365 2212 2110 21198 ...
 - attr(*, "dimnames")=List of 2
  ..$ : chr [1:50] "Alabama" "Alaska" "Arizona" "Arkansas" ...
  ..$ : chr [1:8] "Population" "Income" "Illiteracy" "Life Exp" ...
```

In [15]:
```
nrow(state.x77)
```

50

In [16]:
```
ncol(state.x77)
```

8

In [17]:
```
rownames(state.x77)
```

'Alabama' 'Alaska' 'Arizona' 'Arkansas' 'California' 'Colorado' 'Connecticut' 'Delaware' 'Florida' 'Georgia' 'Hawaii' 'Idaho' 'Illinois' 'Indiana' 'Iowa' 'Kansas' 'Kentucky' 'Louisiana' 'Maine' 'Maryland' 'Massachusetts' 'Michigan' 'Minnesota' 'Mississippi' 'Missouri' 'Montana' 'Nebraska' 'Nevada' 'New Hampshire' 'New Jersey' 'New Mexico' 'New York' 'North Carolina' 'North Dakota' 'Ohio' 'Oklahoma' 'Oregon' 'Pennsylvania' 'Rhode Island' 'South Carolina' 'South Dakota' 'Tennessee' 'Texas' 'Utah' 'Vermont' 'Virginia' 'Washington' 'West Virginia' 'Wisconsin' 'Wyoming'

In [18]:
```
colnames(state.x77)
```

'Population' 'Income' 'Illiteracy' 'Life Exp' 'Murder' 'HS Grad' 'Frost' 'Area'

Indexing and slicing

1) Selecting cells

In [19]:
```
cricket[1,2]
```

10000

In [20]:
```
cricket[1:3,1:2]
```

players	runs
Tendulkar	10000
Kohli	7100
Dohni	5800

In [21]:
```
cricket[c(1,4),c(1,3)]
```

	players	wickets
1	Tendulkar	11
4	Bumrah	370

In [22]:
```
cricket[1:3,'wickets']
```

11 0 0

2) Selecting rows

In [23]:
```
cricket[1,]
```

players	runs	wickets
Tendulkar	10000	11

In [24]:
```
cricket[1:3,]
```

players	runs	wickets
Tendulkar	10000	11
Kohli	7100	0
Dohni	5800	0

In [25]:
```
cricket[c(1,3),]
```

	players	runs	wickets
1	Tendulkar	10000	11
3	Dohni	5800	0

3) Selecting columns

In [26]:
```
cricket[,3]
```

11 0 0 370 420

In [27]:
```
cricket[,1:2]
```

players	runs
Tendulkar	10000
Kohli	7100
Dohni	5800
Bumrah	890
Chahal	870

In [28]:
```
cricket[,c(1,3)]
```

players	wickets
Tendulkar	11
Kohli	0
Dohni	0
Bumrah	370
Chahal	420

In [29]:
```
cricket[,'runs']
```

10000 7100 5800 890 870

In [30]:
```
cricket[,c('players','wickets')]
```

players	wickets
Tendulkar	11
Kohli	0
Dohni	0
Bumrah	370
Chahal	420

In [31]:
```
cricket[['wickets']]
```

11 0 0 370 420

In [32]:
```
cricket$players
```

Tendulkar Kohli Dohni Bumrah Chahal

▶ **Levels**:

Conditional selection with subset() function

In [33]:
```
subset(cricket,subset=runs>5000)
```

players	runs	wickets
Tendulkar	10000	11
Kohli	7100	0
Dohni	5800	0

In [34]:
```
subset(cricket,subset=wickets>300)
```

	players	runs	wickets
4	Bumrah	890	370
5	Chahal	870	420

Ordering dataframe

Ordering is done with order() function

In [35]:
```
order(cricket['runs'])
```

5 4 3 2 1

In [36]:
```
runs.order <- order(cricket['runs'])

cricket[runs.order,]
```

	players	runs	wickets
5	Chahal	870	420
4	Bumrah	890	370
3	Dohni	5800	0
2	Kohli	7100	0
1	Tendulkar	10000	11

In [37]:
```
runs.order <- order(-cricket['runs'])

cricket[runs.order,]
```

players	runs	wickets
Tendulkar	10000	11
Kohli	7100	0
Dohni	5800	0
Bumrah	890	370
Chahal	870	420

Renaming Names

In [38]:
```
mat <- matrix(1:20,nrow=5)
```

In [39]:
```
print(mat)
```

```
     [,1] [,2] [,3] [,4]
[1,]    1    6   11   16
[2,]    2    7   12   17
[3,]    3    8   13   18
[4,]    4    9   14   19
[5,]    5   10   15   20
```

In [40]:
```
df <- data.frame(mat)
```

In [41]:
```
df
```

X1	X2	X3	X4
1	6	11	16
2	7	12	17
3	8	13	18
4	9	14	19
5	10	15	20

In [42]:
```
# renaming single column

colnames(df)[1] <- 'index'
```

In [43]:
```
df
```

index	X2	X3	X4
1	6	11	16
2	7	12	17
3	8	13	18
4	9	14	19
5	10	15	20

In [44]:
```
# renaming multiple column

colnames(df) <- c('A','B','C','D')
```

In [45]:
```
df
```

A	B	C	D
1	6	11	16
2	7	12	17
3	8	13	18
4	9	14	19
5	10	15	20

Adding new rows and columns

In [46]:
```
# Adding row

df2 <- data.frame(A=20,B=34,C=67,D=56)

df <- rbind(df,df2)
```

In [47]:
```
df
```

A	B	C	D
1	6	11	16
2	7	12	17
3	8	13	18
4	9	14	19
5	10	15	20
20	34	67	56

In [48]:
```
# adding columns using replicate function

df$E <- rep(NA,nrow(df))
```

In [49]:
```
df
```

A	B	C	D	E
1	6	11	16	NA
2	7	12	17	NA
3	8	13	18	NA
4	9	14	19	NA
5	10	15	20	NA
20	34	67	56	NA

In [50]:
```
df$F <- 10:15
```

In [51]:
```
df
```

A	B	C	D	E	F
1	6	11	16	NA	10
2	7	12	17	NA	11
3	8	13	18	NA	12
4	9	14	19	NA	13
5	10	15	20	NA	14
20	34	67	56	NA	15

In [52]:
```
#copying another column

df$G = df$E
```

In [53]:
```
df
```

A	B	C	D	E	F	G
1	6	11	16	NA	10	NA
2	7	12	17	NA	11	NA
3	8	13	18	NA	12	NA
4	9	14	19	NA	13	NA
5	10	15	20	NA	14	NA
20	34	67	56	NA	15	NA

In [54]:
```
# cbind function

V <- c(10,20,30,40,50,60)

df <- cbind(df,V)
```

In [55]:
```
df
```

A	B	C	D	E	F	G	V
1	6	11	16	NA	10	NA	10
2	7	12	17	NA	11	NA	20
3	8	13	18	NA	12	NA	30
4	9	14	19	NA	13	NA	40
5	10	15	20	NA	14	NA	50
20	34	67	56	NA	15	NA	60

Handling missing values

check presence of missing values with **any()** and **is.na()** function

In [56]:
```
# Check in specific column

any(is.na(df$A))
```

FALSE

In [57]:
```
any(is.na(df$E))
```

TRUE

In [58]:
```
# Check entire dataframe

any(is.na(df))
```

TRUE

Replacing missing values

In [59]:
```
df$G[is.na(df$G)] <- 0
```

In [60]:
```
df
```

A	B	C	D	E	F	G	V
1	6	11	16	NA	10	0	10
2	7	12	17	NA	11	0	20
3	8	13	18	NA	12	0	30
4	9	14	19	NA	13	0	40
5	10	15	20	NA	14	0	50
20	34	67	56	NA	15	0	60

In [61]:
```
df[is.na(df)] <- -1
```

In [62]:
```
df
```

A	B	C	D	E	F	G	V
1	6	11	16	-1	10	0	10
2	7	12	17	-1	11	0	20
3	8	13	18	-1	12	0	30
4	9	14	19	-1	13	0	40
5	10	15	20	-1	14	0	50
20	34	67	56	-1	15	0	60

Missing data must be replaced with mean, median or mode.