

▼ IMPORTING LIB

```
import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns
```

▼ IMPORTING DATA

```
df = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unclean.xlsx")
```

```
employee_master = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unc]  
workload_demand = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unc]  
Shift_allocation = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unc]  
Timelog_sheet = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unc]  
Overtime_cost = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unc]  
work_capacity = pd.read_excel(r"C:\Users\anush\Downloads\DA\PROJECT\Dataset\wrkforce unc]
```

▼ EXPLORING DATA employee_master

```
employee_master.head()
```

	employee_id	employee_name	department	role	grade_level	hire_date	contract_type
0	1	Amy Moore	Quality	Managered	3	2020-11-20	Part-time
1	2	Nicole Wolf PhD	Quality	Senior Executive	4	2021-04-07	Part-time
2	3	Mario Montgomery**	Logisticsed	Executive	5	2020-12-01	Full-time
3	4	Wendy //Green	Production	Manager	8	2025-07-17	Part-time
4	5	Theresa Lewis	Quality	Manager	9	2020-06-11	Contract

```
employee_master.tail()
```

	employee_id	employee_name	department	role	grade_level	hire_date	contract_type
1995	1996	Erika Moss	Logistics	Technician	3	2025-04-28	Contract
1996	1997	Tamara Bell	Production	Technician	8	2020-10-21	Part-time
1997	1998	Michael Reed	Back-office	Analyst	2	2020-11-06	Full-time
1998	1999	Richard George	Quality	Senior Executive	5	2024-06-17	Contract
1999	2000	Mrs. Elizabeth Lambert	Quality	Analyst	1	2022-09-21	Part-time

```
employee_master.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   employee_id      2000 non-null   int64  
 1   employee_name    2000 non-null   object  
 2   department        2000 non-null   object  
 3   role              2000 non-null   object  
 4   grade_level       2000 non-null   int64  
 5   hire_date         2000 non-null   datetime64[ns]
 6   contract_type    2000 non-null   object  
 7   base_hourly_rate 2000 non-null   int64  
 8   overtime_rate    2000 non-null   int64  
 9   max_weekly_hours 2000 non-null   int64  
 10  location          2000 non-null   object  
 11  status             2000 non-null   object  
dtypes: datetime64[ns](1), int64(5), object(6)
memory usage: 187.6+ KB
```

```
employee_master.describe()
```

	employee_id	grade_level	hire_date	base_hourly_rate	overtime_rate	max_weekly_hours
count	2000.000000	2000.000000	2000	2000.000000	2000.000000	2000.000000
mean	1000.500000	4.987000	2022-12-03 09:48:57.600000	327.086000	524.665000	46.000000
min	1.000000	1.000000	2019-12-02 00:00:00	150.000000	300.000000	35.000000
25%	500.750000	3.000000	2021-06-15 18:00:00	238.750000	409.750000	41.000000
50%	1000.500000	5.000000	2022-12-09 12:00:00	328.000000	528.000000	47.000000
75%	1500.250000	7.000000	2024-05-16 18:00:00	412.000000	636.000000	53.000000
max	2000.000000	9.000000	2025-11-30 00:00:00	499.000000	749.000000	59.000000
std	577.494589	2.569851	Nan	100.425511	129.294939	7.000000

```
employee_master.columns
```

```
Index(['employee_id', 'employee_name', 'department', 'role', 'grade_level',
       'hire_date', 'contract_type', 'base_hourly_rate', 'overtime_rate',
       'max_weekly_hours', 'location', 'status'],
      dtype='object')
```

```
employee_master.shape
```

```
(2000, 12)
```

DATA MANIPULATION

```
employee_master.isnull().sum()
```

```
employee_id      0
employee_name    0
department       0
role             0
grade_level      0
hire_date        0
contract_type    0
base_hourly_rate 0
overtime_rate    0
max_weekly_hours 0
location         0
status           0
dtype: int64
```

```
employee_master.duplicated().sum()
```

```
np.int64(0)
```

```
employee_master['employee_name'] = employee_master['employee_name'].astype(str).replace({r'
```

```
employee_master['employee_name'].head(5)
```

```
0      Amy Moore
1      Nicole Wolf PhD
2      Mario Montgomery
3      Wendy Green
4      Theresa Lewis
Name: employee_name, dtype: object
```

```
employee_master['department'] = employee_master['department'].replace("Logisticsed", "Logis1
```

```
employee_master['department']
```

```
0      Quality
1      Quality
2      Logistics
3      Production
4      Quality
...
1995    Logistics
1996    Production
1997    Back-office
1998    Quality
```

```
1999      Quality
Name: department, Length: 2000, dtype: object
```

```
employee_master['role'] = employee_master['role'].replace("Managered","Manager")
```

EXPLORING DATA workload_demand

```
workload_demand.head()
```

	demand_id	date	department	forecasted_hours_required	actual_hours_required	peak_flag
0	1	2024-08-30	Production	103	904	No
1	2	2024-11-04	Back-office	464	863	Yes
2	3	2024-07-24	Quality	391	476	No
3	4	2024-11-11	Sales	419	448	Yes
4	5	2024-10-20	Production	230	217	No

```
workload_demand.tail()
```

	demand_id	date	department	forecasted_hours_required	actual_hours_required	peak_flag
995	996	2024-07-18	Sales	862	235	I
996	997	2024-12-23	Quality	627	541	Y
997	998	2024-05-21	Sales	322	513	I
998	999	2024-02-07	Production	682	336	I
999	1000	2024-12-14	Quality	637	498	I

```
workload_demand.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   demand_id        1000 non-null   int64  
 1   date             1000 non-null   datetime64[ns]
 2   department       1000 non-null   object  
 3   forecasted_hours_required  1000 non-null   int64  
 4   actual_hours_required    1000 non-null   int64  
 5   peak_flag         1000 non-null   object  
 6   demand_source     1000 non-null   object  
 7   werroe            1000 non-null   object  
dtypes: datetime64[ns](1), int64(3), object(4)
memory usage: 62.6+ KB
```

```
workload_demand.describe()
```

	demand_id	date	forecasted_hours_required	actual_hours_required
count	1000.000000	1000	1000.000000	1000.000000
mean	500.500000	2024-06-30 18:46:04.799999744	499.325000	518.744000
min	1.000000	2024-01-01 00:00:00	100.000000	80.000000
25%	250.750000	2024-03-30 18:00:00	292.000000	300.500000
50%	500.500000	2024-06-26 00:00:00	498.000000	520.000000
75%	750.250000	2024-10-04 00:00:00	701.000000	746.000000
max	1000.000000	2024-12-31 00:00:00	899.000000	949.000000
std	288.819436	Nan	234.405775	250.192855

```
workload_demand.shape
```

```
(1000, 8)
```

```
workload_demand.columns
```

```
Index(['demand_id', 'date', 'department', 'forecasted_hours_required',
       'actual_hours_required', 'peak_flag', 'demand_source', 'werroe'],
      dtype='object')
```

DATA MANIPULATION

```
workload_demand.isnull().sum()
```

```
demand_id          0
date              0
department        0
forecasted_hours_required  0
actual_hours_required  0
peak_flag          0
demand_source      0
werroe             0
dtype: int64
```

```
workload_demand.duplicated().sum()
```

```
np.int64(0)
```

```
workload_demand['department'] = (
    workload_demand['department']
    .astype(str)
    .str.replace(r"[\*/\-\-]", "", regex=True)
)
```

```
workload_demand['demand_source'] = workload_demand['demand_source'].replace({'Retail':'Retail'})
```

```
workload_demand = workload_demand.drop(columns=['werroe'])
```

workload_demand

	demand_id	date	department	forecasted_hours_required	actual_hours_required	peak_flag
0	1	2024-08-30	Production		103	I
1	2	2024-11-04	Backoffice		464	Y
2	3	2024-07-24	Quality		391	I
3	4	2024-11-11	Sales		419	Y
4	5	2024-10-20	Production		230	I
...
995	996	2024-07-18	Sales		862	I
996	997	2024-12-23	Quality		627	Y
997	998	2024-05-21	Sales		322	I
998	999	2024-02-07	Production		682	I
999	1000	2024-12-14	Quality		637	I

1000 rows × 7 columns

```
workload_demand['date'] = pd.to_datetime(workload_demand['date'], errors='coerce')
```

▼ EXPLORING DATA - shift_allocation

Shift_allocation.head()

	shift_id	employee_id	date	shift_start	shift_end	planned_hours	shift_type	location
0	1.0	884	2024-12-07	00:00:00	08:00	17:00	9	Evening Jacob
1	2.0	952	2024-11-07	00:00:00	08:00	17:00	8	Evening New Cy
2	NaN	*	*	*	*	*	*	*
3	NaN	*	*	*	*	*	*	*
4	3.0	1676	2024-01-07	00:00:00	08:00	17:00	8	Morning Gonzalez

```
Shift_allocation.tail()
```

	shift_id	employee_id	date	shift_start	shift_end	planned_hours	shift_type	location
4997	4996.0	1116	2024-03-23 00:00:00	08:00	17:00	8	Morning	Sou
4998	4997.0	1923	2024-10-05 00:00:00	08:00	17:00	9	Morning	East
4999	4998.0	1003	2024-08-04 00:00:00	08:00	17:00	9	Night	Gar
5000	4999.0	1881	2024-04-27 00:00:00	08:00	17:00	8	Weekend	Med
5001	5000.0	1481	2024-05-09 00:00:00	08:00	17:00	7	Weekend	West

```
Shift_allocation.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5002 entries, 0 to 5001
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype  
--- 
 0   shift_id      5000 non-null   float64
 1   employee_id   5002 non-null   object  
 2   date          5002 non-null   object  
 3   shift_start    5002 non-null   object  
 4   shift_end      5002 non-null   object  
 5   planned_hours  5002 non-null   object  
 6   shift_type     5002 non-null   object  
 7   location       5002 non-null   object  
dtypes: float64(1), object(7)
memory usage: 312.8+ KB
```

```
Shift_allocation.describe()
```

	shift_id
count	5000.000000
mean	2500.500000
std	1443.520003
min	1.000000
25%	1250.750000
50%	2500.500000
75%	3750.250000
max	5000.000000

```
Shift_allocation.columns
```

```
Index(['shift_id', 'employee_id', 'date', 'shift_start', 'shift_end',
       'planned_hours', 'shift_type', 'location'],
```

```
        dtype='object')
```

```
Shift_allocation.shape
```

```
(5002, 8)
```

DATA MANIPULATION

```
Shift_allocation = Shift_allocation.astype(str).replace({r"[*/\-\"]": " "}, regex=True)
Shift_allocation = Shift_allocation.replace("nan", np.nan)
Shift_allocation = Shift_allocation.dropna(how="all")
```

```
Shift_allocation.isnull().sum()
```

```
shift_id      2
employee_id   0
date          0
shift_start   0
shift_end     0
planned_hours 0
shift_type    0
location      0
dtype: int64
```

```
Shift_allocation = Shift_allocation.dropna()
```

```
Shift_allocation.isnull().sum()
```

```
shift_id      0
employee_id   0
date          0
shift_start   0
shift_end     0
planned_hours 0
shift_type    0
location      0
dtype: int64
```

```
Shift_allocation['date'] = pd.to_datetime(Shift_allocation['date'], errors='coerce')
```

```
Shift_allocation.head()
```

	shift_id	employee_id	date	shift_start	shift_end	planned_hours	shift_type	location
0	1.0	884	2024-12-07	08:00	17:00	9	Evening	Jacobsh
1	2.0	952	2024-11-07	08:00	17:00	8	Evening	New Cynth
4	3.0	1676	2024-01-07	08:00	17:00	8	Morning	Gonzalezf
5	4.0	1897	2024-11-20	08:00	17:00	7	Weekend	Lan
6	5.0	1027	2024-07-24	08:00	17:00	7	Morning	Torresbur

EXPLORING DATA -Timelog_sheet

Timelog_sheet.head()

	timesheet_id	employee_id	date	planned_hours	actual_hours	overtime_hours	apps_by
0	1.0	123.0	2024-04-22		8.0	12.0	4.0 Eric Turner
1	2.0	448.0	2024-06-29		8.0	7.0	0.0 Andrew Barton
2	3.0	481.0	2024-01-05		8.0	5.0	5.0 Dr. Diana Brown
3	4.0	1172.0	2024-01-28		7.0	10.0	2.0 Debbie Rogers
4	5.0	1712.0	2024-12-15		9.0	5.0	4.0 Noah Thompson

Timelog_sheet.tail()

	timesheet_id	employee_id	date	planned_hours	actual_hours	overtime_hours	apps_by
9998	9996.0	226.0	2024-07-03		7.0	6.0	3.0 Taylor Wang
9999	9997.0	942.0	2024-12-08		9.0	11.0	3.0 Timothy Myer
10000	9998.0	1252.0	2024-08-13		9.0	8.0	4.0 Jamie Gibb
10001	9999.0	1074.0	2024-11-11		8.0	12.0	2.0 Corey King
10002	10000.0	1991.0	2024-03-04		7.0	8.0	0.0 Tiffany Dura

Timelog_sheet.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10003 entries, 0 to 10002
Data columns (total 9 columns):
```

```
# Column      Non-Null Count Dtype
--- 
0  timesheet_id    10000 non-null   float64
1  employee_id     10000 non-null   float64
2  date           10000 non-null   datetime64[ns]
3  planned_hours   10000 non-null   float64
4  actual_hours    10000 non-null   float64
5  overtime_hours  10000 non-null   float64
6  apps_by         10000 non-null   object
7  approval_status 10000 non-null   object
8  task_type        10000 non-null   object
dtypes: datetime64[ns](1), float64(5), object(3)
memory usage: 703.5+ KB
```

```
Timelog_sheet.describe()
```

	timesheet_id	employee_id	date	planned_hours	actual_hours	overtime_hours
count	10000.00000	10000.00000		10000	10000.00000	10000.00000
mean	5000.50000	1003.429700	2024-06-30 10:57:47.520000256		8.011800	8.988700
min	1.00000	1.000000	2024-01-01 00:00:00		7.000000	5.000000
25%	2500.75000	504.000000	2024-03-31 00:00:00		7.000000	7.000000
50%	5000.50000	1001.000000	2024-06-30 00:00:00		8.000000	9.000000
75%	7500.25000	1509.000000	2024-09-28 00:00:00		9.000000	11.000000

```
Timelog_sheet.shape
```

```
(10003, 9)
```

```
Timelog_sheet.columns
```

```
Index(['timesheet_id', 'employee_id', 'date', 'planned_hours', 'actual_hours',
       'overtime_hours', 'apps_by', 'approval_status', 'task_type'],
      dtype='object')
```

DATA MANIPULATION

```
Timelog_sheet.isnull().sum()
```

```
timesheet_id      3
employee_id      3
date             3
planned_hours    3
actual_hours     3
overtime_hours   3
apps_by          3
approval_status  3
task_type         3
dtype: int64
```

```
Timelog_sheet = Timelog_sheet.dropna()
```

```
Timelog_sheet.isnull().sum()
```

	0
timesheet_id	0
employee_id	0
date	0
planned_hours	0
actual_hours	0
overtime_hours	0
apps_by	0
approval_status	0
task_type	0
dtype: int64	

```
Timelog_sheet.duplicated().sum()
```

```
np.int64(0)
```

```
Timelog_sheet['approval_status'].unique()
```

```
array(['apps', 'Pending', 'Rejected'], dtype=object)
```

```
Timelog_sheet['approval_status'] = Timelog_sheet['approval_status'].replace("apps", "Approved")
```

```
Timelog_sheet['task_type'].unique()
```

```
array(['Admitted', 'Admin', 'Production', 'QA', 'Support'], dtype=object)
```

```
Timelog_sheet['task_type'] = Timelog_sheet['task_type'].replace({"Admitted": "Admin", "Production": "Production", "QA": "Quality Assurance", "Support": "Customer Support", "Admin": "Management", "Admitted": "Admission"}))
```

EXPLORING DATA - Overtime_cost

```
Overtime_cost.head()
```

	overtime_id	employee_id	date	overtime_hours	overtime_rate	overtime_cost	reason	apps
0	1	361	2024-08-26		5	548	2740	staff sort
1	2	433	2024-05-21		2	410	820	staff sort
2	3	1005	2024-01-13		4	399	1596	staff sort
3	4	1146	2024-09-05		1	521	521	Urgent Task
4	5	424	2024-05-24		2	737	1474	Shift Gap

```
Overtime_cost.tail()
```

	overtime_id	employee_id	date	overtime_hours	overtime_rate	overtime_cost	reason
7995	7996	1779	2024-10-19	4	783	3132	staff shortage
7996	7997	1739	2024-05-30	1	359	359	Urgent Task
7997	7998	1578	2024-11-28	1	774	774	Workload Spike
7998	7999	319	2024-02-28	1	677	677	Shift Gap
7999	8000	1555	2024-07-14	1	587	587	Shift Gap

Overtime_cost.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8000 entries, 0 to 7999
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   overtime_id      8000 non-null    int64  
 1   employee_id      8000 non-null    int64  
 2   date              8000 non-null    datetime64[ns]
 3   overtime_hours    8000 non-null    int64  
 4   overtime_rate     8000 non-null    int64  
 5   overtime_cost     8000 non-null    int64  
 6   reason             8000 non-null    object  
 7   approved_by        8000 non-null    object  
dtypes: datetime64[ns](1), int64(5), object(2)
memory usage: 500.1+ KB
```

Overtime_cost.describe()

	overtime_id	employee_id	date	overtime_hours	overtime_rate	overtime_cost	
count	8000.00000	8000.000000		8000	8000.000000	8000.000000	
mean	4000.50000	999.082625	2024-07-02 22:59:31.200000256		2.997375	574.240750	1721.50
min	1.00000	1.000000	2024-01-01 00:00:00		1.000000	350.000000	350.00
25%	2000.75000	492.000000	2024-04-02 00:00:00		2.000000	461.000000	918.00
50%	4000.50000	1010.000000	2024-07-03 00:00:00		3.000000	575.000000	1587.00
75%	6000.25000	1496.250000	2024-10-04 00:00:00		4.000000	687.000000	2346.00

Overtime_cost.columns

```
Index(['overtime_id', 'employee_id', 'date', 'overtime_hours', 'overtime_rate',
       'overtime_cost', 'reason', 'approved_by'],
      dtype='object')
```

Overtime_cost.shape

```
(8000, 8)
```

▼ DATA MANIPULATION

```
Overtime_cost.isnull().sum()
```

```
overtime_id      0  
employee_id     0  
date            0  
overtime_hours  0  
overtime_rate   0  
overtime_cost   0  
reason          0  
approved_by     0  
dtype: int64
```

```
Overtime_cost.duplicated().sum()
```

```
np.int64(0)
```

```
Overtime_cost['reason'] = Overtime_cost['reason'].replace('staff sort','staff shortage')
```

▼ DATA EXPLORING

```
work_capacity.head()
```

	capacity_id	date	department	total_employees_available	total_planned_hours	total_act
0	1	2024-09-15	Logistics		27	479
1	2	2024-07-05	Back-office		18	565
2	3	2024-03-10	Quality		98	809
3	4	2024-12-14	Quality		29	785
4	5	2024-06-07	Back-office		13	831

```
work_capacity.tail()
```

	capacity_id	date	department	total_employees_available	total_planned_hours	total
1212	112	2024-03-08	Customer Support	35	656	
1213	113	2024-06-30	Sales	25	619	
1214	114	2024-06-29	Customer Support	13	744	
1215	115	2024-07-25	Back-office	44	353	
1216	116	2024-11-08	Sales	29	865	

```
work_capacity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1217 entries, 0 to 1216
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   capacity_id      1217 non-null   int64  
 1   date              1217 non-null   datetime64[ns]
 2   department        1217 non-null   object  
 3   total_employees_available  1217 non-null   int64  
 4   total_planned_hours  1217 non-null   int64  
 5   total_actual_hours  1217 non-null   int64  
 6   total_overtime_hours  1217 non-null   int64  
 7   capacity_gap      1217 non-null   int64  
 8   utilization_rate  1217 non-null   float64 
 9   overtime_dependency_score  1217 non-null   float64 
dtypes: datetime64[ns](1), float64(2), int64(6), object(1)
memory usage: 95.2+ KB
```

```
work_capacity.describe()
```

	capacity_id	date	total_employees_available	total_planned_hours	total
count	1217.000000	1217	1217.000000	1217.000000	1217.000000
mean	593.620378	2024-07-03 20:29:23.023828992	78.364832	549.156122	
min	1.000000	2024-01-01 00:00:00	10.000000	200.000000	
25%	288.000000	2024-04-07 00:00:00	42.000000	385.000000	
50%	592.000000	2024-07-04 00:00:00	78.000000	549.000000	
75%	896.000000	2024-10-01 00:00:00	113.000000	717.000000	
max	1200.000000	2024-12-31 00:00:00	149.000000	898.000000	
std	348.948331	Nan	41.200289	199.044086	

```
work_capacity.shape
```

```
(1217, 10)
```

```
work_capacity.columns
```