# Manufacturing Downtime Analysis Report

## 1. Introduction

This project focuses on analyzing **manufacturing machine performance** and understanding the causes of **downtime** by using industrial sensor data.
The dataset — titled *"Manufacturing Downtime Dataset"* — contains **2,500 rows and 16 columns**, representing sensor readings collected from multiple machines operating across different assembly lines.

The main goals of the project are:

- To **clean** and **structure** the dataset properly for analysis.
- To **build a relational SQL database** that organizes machine data efficiently.
- To **define and calculate KPIs** that reflect machine performance and production efficiency.
- To prepare the data for **future predictive modeling** and **dashboard visualization**.

---

## 2. Dataset Overview

The dataset contains a combination of **machine IDs**, **line numbers**, **dates**, and multiple **sensor readings** that describe the machine's operational state at a given time.

**Dataset Dimensions**

- **Rows:** 2,500
- **Columns:** 16

**Main Features**

| Column Name | Description | Importance |
| --- | --- | --- |
| **Machine_ID** | Unique identifier for each machine | Used to track performance of each individual machine |
| **Assembly_Line_No** | The production line where the machine operates | Helps analyze efficiency per line |
| **Date** | The date of measurement | Allows time-based analysis |
| **Hydraulic_Pressure (bar)** | Fluid pressure within the hydraulic system | Detects mechanical efficiency and fluid issues |
| **Coolant_Pressure (bar)** | Pressure in the cooling system | Detects coolant leaks or pump failures |
| **Air_System_Pressure (bar)** | Air pressure in the pneumatic system | Ensures stability in air-driven tools |
| **Coolant_Temperature (°C)** | Temperature of the coolant | Detects potential overheating |
| **Hydraulic_Oil_Temperature (°C)** | Temperature of hydraulic oil | Reflects lubrication and maintenance needs |
| **Spindle_Bearing_Temperature (°C)** | Temperature of spindle bearings | Early warning of wear or bearing failure |
| **Spindle_Vibration (µm)** | Vibration of the spindle | Measures mechanical balance |
| **Tool_Vibration (µm)** | Tool vibration level | Affects cutting accuracy and quality |

| Column Name | Description | Importance |
|---|---|---|
| **Spindle_Speed (RPM)** | Rotation speed of the spindle | Indicates productivity and process speed |
| **Voltage (volts)** | Power supply voltage | Detects electrical instability |
| **Torque (Nm)** | Rotational force of the spindle | Indicates energy consumption and resistance |
| **Cutting (kN)** | Cutting force during operation | Reflects process intensity |
| **Downtime_Status** | Indicates whether the machine was down or active | Used to calculate availability and downtime KPIs |

---

# 3. Data Cleaning and Preprocessing

Data cleaning was a crucial stage to ensure data accuracy and consistency before any analysis.
This process was done in **Python (Pandas)** and **SQL**, following these detailed steps:

---

### Step 1: Loading the Dataset

```
import numpy as np
import pandas as pd
df = pd.read_csv("/content/Manufacturing_Downtime_Dataset_(1)(1).csv")
```

After loading, we confirmed the shape of the dataset:

```
df.shape
```

☑ Output: **(2500, 16)** — confirming 2,500 rows and 16 columns.

---

### Step 2: Checking for Missing Values

We used:

```
df.isna().sum()
```

to identify columns with missing values.
Then calculated missing value percentages:

```
missing_percentage = ((df.isna().sum() / len(df)) * 100).round(3)
```

**Findings:**

- Minor missing values (1–5%) were found in numerical columns like pressure, temperature, and vibration.
- No missing categorical values (e.g., machine IDs or dates).

---

### Step 3: Handling Missing Values

To maintain data consistency, missing numerical values were replaced with their **mean**:

```
df['Hydraulic_Pressure(bar)'] =
df['Hydraulic_Pressure(bar)'].fillna(df['Hydraulic_Pressure(bar)'].mea
n())
df['Coolant_Temperature'] =
df['Coolant_Temperature'].fillna(df['Coolant_Temperature'].mean())
df['Torque(Nm)'] = df['Torque(Nm)'].fillna(df['Torque(Nm)'].mean())
# Repeated for all numerical columns
```

---

### Step 4: Removing Duplicates

```
df = df.drop_duplicates()
```

This guarantees that each record represents a **unique machine reading** at a specific time.

### Step 5: SQL Structuring and Relationships

After cleaning, the data was imported into a **MySQL database** called `manufacturing_project`.

The SQL structure was organized into **three related tables** for better data integrity:

1. **machines_data** → stores machine IDs and assembly line numbers
2. **reading_dates** → stores unique reading dates
3. **machine_readings** → contains all sensor readings, linked by foreign keys

*Foreign Key Relationships*

```
FOREIGN KEY (machine_id) REFERENCES machines_data(machine_id),
FOREIGN KEY (assembly_line_no) REFERENCES
machines_data(assembly_line_no),
FOREIGN KEY (date_id) REFERENCES reading_dates(date_id)
```

This normalization prevents redundancy and improves data querying speed for analysis and reporting.

---

# 4. KPI (Key Performance Indicators)

After the dataset was cleaned and structured, we defined key **performance metrics** (KPIs) to evaluate machine and production efficiency.

| KPI Name | Formula / Method | Purpose |
|---|---|---|
| **Average Downtime (%)** | `(Total Downtime Machines / Total Machines) × 100` | Measures machine availability |
| **Mean Hydraulic Pressure (bar)** | `mean(Hydraulic_Pressure)` | Reflects the average fluid system performance |

| KPI Name | Formula / Method | Purpose |
| --- | --- | --- |
| **Average Coolant Temperature (°C)** | `mean(Coolant_Temperature)` | Tracks heat regulation effectiveness |
| **Average Torque (Nm)** | `mean(Torque)` | Shows how much effort machines exert during operation |
| **Mean Spindle Speed (RPM)** | `mean(Spindle_Speed)` | Indicates the general speed of operation |
| **Average Vibration (µm)** | `(mean(Spindle_Vibration) + mean(Tool_Vibration)) / 2` | Detects machine stability |
| **Line Efficiency (%)** | `(Active Machines per Line / Total Machines per Line) × 100` | Evaluates performance by assembly line |
| **Power Stability (voltage variance)** | `std(Voltage)` | Monitors electrical system reliability |

# 5. Results and Insights

- After cleaning, the dataset had **no missing values** and **no duplicate entries**.
- The structure supports efficient querying for each machine, line, or date.
- Mean values for pressure and temperature columns show balanced system operation.
- Vibration KPIs will be key indicators for **predictive maintenance** and **failure prediction**.

# 6. Next Steps

1. **Predictive Modeling:**
   Train models to predict machine downtime based on sensor patterns (e.g., Logistic Regression, Random Forest).
2. **Dashboard Visualization:**
   Use Power BI or Looker Studio to create an **interactive dashboard** that shows KPIs and machine performance trends in real time.
3. **Anomaly Detection:**
   Implement algorithms to automatically flag unusual sensor readings that might signal early-stage machine failure.

---

# 7. Conclusion

This project successfully:

- Cleaned and standardized a complex manufacturing dataset.
- Organized the data into a **normalized SQL schema** with strong relationships.
- Defined meaningful **KPIs** for performance monitoring.

The cleaned and structured data is now **ready for analysis, visualization, and prediction**, allowing manufacturers to take **data-driven actions** that reduce downtime and improve operational efficiency