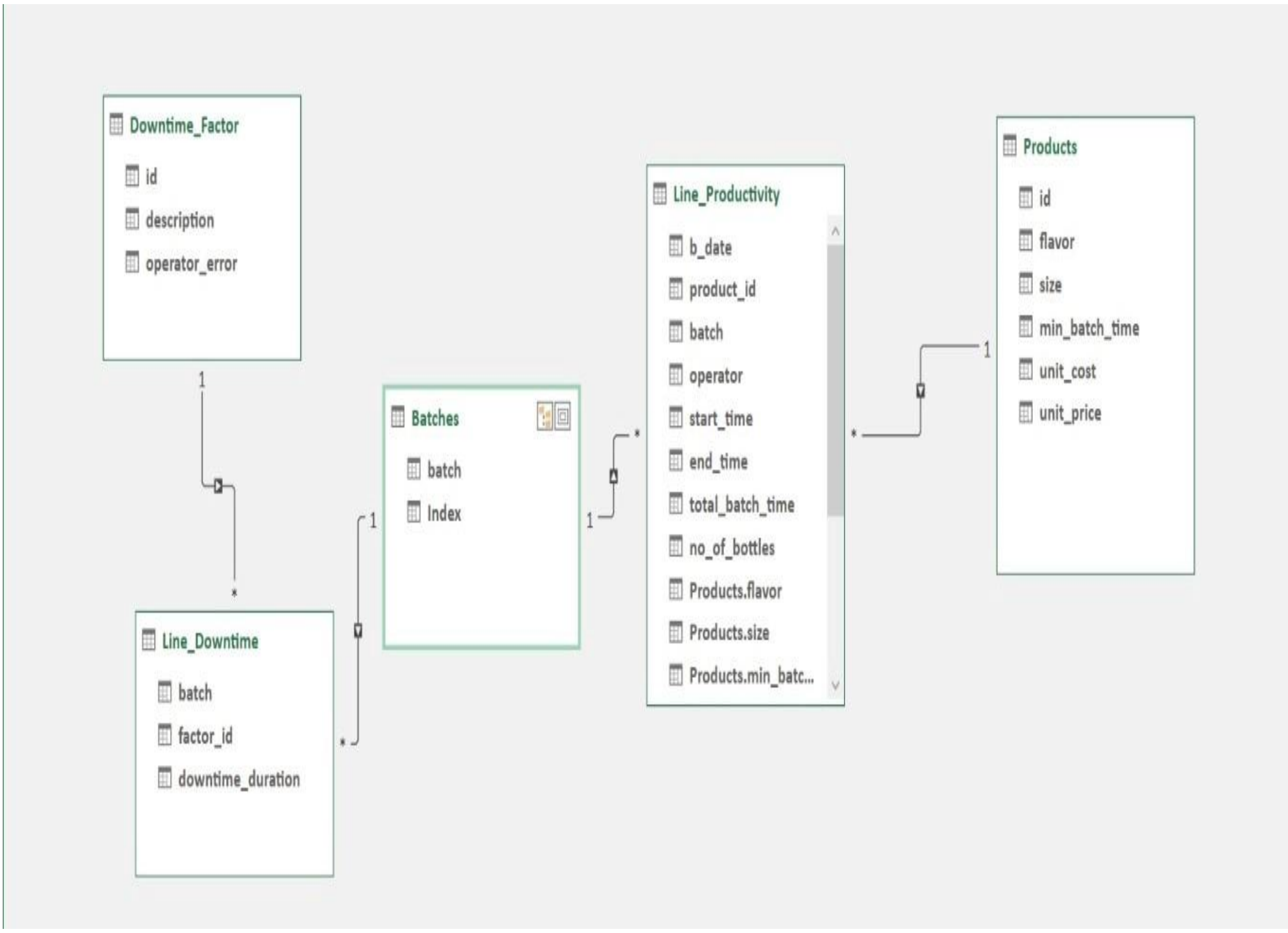
DATA WIZARDS
CASTING INSIGHTS, CRAFTING SOLUTIONS

TECHNICAL REPORT

- ## Data cleaning and modeling:

done by using Microsoft Excel query: For initial data cleaning, organization, and basic analysis.



SQL SYNTAX for handling and answering business questions.

Analytical questions

1. Productivity & Efficiency:

a- What is the average time taken to complete a batch for each product?

```
WITH Total_Production_time AS (
SELECT product_id
      ,Batch
      ,SUM(total_batch_time) AS Total_time
FROM line_productivity
GROUP BY 1, 2
)
,
No_of_batches AS (
SELECT product_id
      ,COUNT(Batch) as batches
FROM line_productivity
GROUP BY product_id
)

SELECT Total_Production_time.product_id
      ,Total_Production_time.Batch
      ,SUM(Total_Production_time.Total_time / No_of_batches.batches) as
Average_time_per_batch
FROM Total_Production_time , No_of_batches
GROUP BY 1, 2
```

The result

product	batch	average_time_per_batch
RB-600	422137	191
CO-2000	422145	219
CO-2000	422146	290
LE-600	422114	181
DC-600	422134	199
CO-600	422123	241
CO-2000	422147	373
DC-600	422135	191
RB-600	422141	120
RB-600	422143	212
CO-600	422120	203

b- Which products have the highest production rate, and which ones consistently take longer to produce?

```
WITH production_rates AS (
   SELECT lp.product_id,
         SUM(lp.actual_no_of_bottles) AS total_units_produced,
         SUM(lp.actual_batch_time + COALESCE(ld.downtime_duration, 0)) AS
total_production_time,
         SUM(lp.actual_no_of_bottles) / SUM(lp.actual_batch_time +
COALESCE(ld.downtime_duration, 0)) AS production_rate
   FROM line_productivity lp
   LEFT JOIN line_downtime ld ON lp.batch = ld.batch
   GROUP BY lp.product_id
)
```

```sql
SELECT product_id,
       total_units_produced,
       total_production_time,
       production_rate
FROM production_rates
ORDER BY production_rate DESC;
```

The result

product_id	total_units_produced	total_production_time	production_rate
LE-600	108000	709	152
DC-600	72000	475	151
CO-600	300000	1994	150
RB-600	132000	918	143
OR-600	24000	195	123
CO-2000	107800	1355	79

2. Operator Performance:

a- Which operators are processing the most batches, and how does their productivity compare?

```sql
SELECT operator,
       COUNT(batch) AS total_batches_processed,
       round(AVG(total_batch_time), 2) AS avg_total_cycle_time,
       round(AVG(actual_batch_time), 2) AS avg_actual_cycle_time,
       SUM(no_of_bottles) AS total_units_produced
FROM line_productivity
GROUP BY operator
ORDER BY total_batches_processed DESC;
```

The result

operator	batches_processed	avg_total_time	avg_actual_time	units_produced
Dee	11	93.64	60	206000
Charlie	11	105.27	70.36	183100
Dennis	8	102.5	64.75	148800
Mac	8	106.25	64.75	157000

b- Are certain operators more efficient with specific products?

```
SELECT operator,
       product_id,
       round(AVG(total_batch_time), 2) AS avg_cycle_time,
       SUM(actual_no_of_bottles) AS total_units_produced
FROM line_productivity
GROUP BY operator, product_id
ORDER BY product_id, avg_cycle_time DESC;
```

The result

operator	product_id	avg_cycle_time	total_units_produced
Charlie	CO-2000	161.67	29400
Dennis	CO-2000	152	9800
Mac	CO-2000	130	9800
Dennis	CO-600	98.25	48000
Dee	CO-600	91.17	72000
Charlie	CO-600	90.8	60000
Mac	DC-600	91.67	36000
Dee	DC-600	80	12000
Mac	LE-600	103.33	36000
Charlie	LE-600	73	36000
Mac	OR-600	135	12000
Dee	RB-600	100.75	48000
Dennis	RB-600	91.67	36000

c- How does downtime affect the performance of different operators?

```
SELECT lp.operator,
       COUNT(ld.factor_id) AS downtime_occurrences,
       SUM(ld.downtime_duration) AS total_downtime_duration,
       COUNT(lp.batch) AS total_batches_processed,
       round(AVG(lp.total_batch_time), 2) AS avg_cycle_time,
       SUM(lp.actual_no_of_bottles) AS total_units_produced
FROM line_productivity lp
LEFT JOIN line_downtime ld ON lp.batch = ld.batch
GROUP BY lp.operator
ORDER BY total_downtime_duration DESC;
```

The result

operator	downtime_occurrences	total_downtime_duration	batches_processed	avg_cycle_time	units_produced
Charlie	17	384	18	119.72	200600
Dee	19	370	20	101.35	240000
Mac	13	332	14	109.29	163600
Dennis	12	302	12	110.25	139600

d- How does operator performance vary across different shifts?

```
SELECT operator,
       CASE
         WHEN EXTRACT(HOUR FROM start_time) BETWEEN 6 AND 14 THEN
'Morning'
         WHEN EXTRACT(HOUR FROM start_time) BETWEEN 14 AND 22 THEN
'Afternoon'
         ELSE 'Night'
```

```
    END AS shift,
    COUNT(batch) AS total_batches_processed,
    round(AVG(total_batch_time),2) AS avg_cycle_time,
    SUM(no_of_bottles) AS total_units_produced
FROM line_productivity
GROUP BY operator, shift
ORDER BY operator, shift;
```

The result

operator	shift	total_batches_processed	avg_cycle_time	total_units_produced
Charlie	Afternoon	10	103.8	171100
Charlie	Morning	1	120	12000
Dee	Morning	6	84.5	101400
Dee	Night	5	104.6	104600
Dennis	Morning	8	102.5	148800
Mac	Afternoon	4	100	67000
Mac	Morning	4	112.5	90000

3. Downtime Analysis:

a- What are the top five factors contributing to downtime?

```
SELECT df.description AS Downtime_cause
    ,SUM(ld.downtime_duration) AS total_downtime
    ,count(*) AS occurrences
FROM line_downtime ld
JOIN downtime_factors df
ON ld.factor_id = df.id
GROUP BY df.description
HAVING count(*) >=5
ORDER BY total_downtime DESC
LIMIT 5;
```

The result

Downtime_cause	total_downtime	occurrences
Machine adjustment	332	12
Machine failure	254	11
Inventory shortage	225	9
Batch change	160	5
Batch coding error	145	6

b- What is the percentage of downtime to total production time?

```
SELECT
  SUM(ld.downtime_duration) AS total_downtime,
  SUM(lp.total_batch_time) AS total_time,
  round(((SUM(ld.downtime_duration) / SUM(lp.total_batch_time)::numeric) *
100), 2) AS downtime_percentage
FROM line_downtime ld
JOIN line_productivity lp ON ld.batch = lp.batch
```

The result

total_downtime	total__Production_time	downtime_percentage
1388	6855	20.25%

c- Is downtime distributed evenly across different products and production lines?

```
SELECT lp.product_id,
       SUM(ld.downtime_duration) AS total_downtime,
       COUNT(ld.batch) AS occurrences,
       ROUND(AVG(ld.downtime_duration), 2) AS avg_downtime
FROM line_downtime ld
JOIN line_productivity lp ON ld.batch = lp.batch
GROUP BY lp.product_id
ORDER BY total_downtime DESC;
```

The result

product	total_downtime	occurrences	avg_downtime
CO-600	494	24	20.58
CO-2000	277	11	25.18
RB-600	258	11	23.45
LE-600	169	8	21.13
DC-600	115	5	23
OR-600	75	2	37.5

3. Financial Impact:

a- What is the financial cost of downtime for each product or line?

```sql
SELECT pc.flavor AS Product
      ,SUM(ld.downtime_duration * (pc.unit_price - pc.unit_cost)) AS
downtime_cost
FROM line_downtime ld
JOIN line_productivity lp ON ld.batch = lp.batch
JOIN products pc ON lp.product_id = pc.id
GROUP BY pc.flavor
ORDER BY downtime_cost DESC;
```

The result

product	downtime_cost
Cola	253.136
Root Berry	64.5
Lemon lime	42.25
Diet Cola	28.75
Orange	18.75

b- How much potential revenue loss due to downtime, and which downtimes are the costliest?

```
SELECT pc.id As product,
       pc.flavor,
       df.description AS downtime_cause,
       SUM(ld.downtime_duration * pc.unit_price) AS potential_revenue_loss
FROM line_downtime ld
JOIN line_productivity lp ON ld.batch = lp.batch
JOIN products pc ON lp.product_id = pc.id
JOIN downtime_factors df ON ld.factor_id = df.id
GROUP BY pc.id, pc.flavor, df.description
ORDER BY pc.flavor, potential_revenue_loss DESC;
```

product	flavor	downtime_cause	potential_revenue_loss	
CO-2000	Cola	Machine adjustment	£	280.80
CO-600	Cola	Machine failure	£	145.00
CO-600	Cola	Inventory shortage	£	135.00
CO-2000	Cola	Machine failure	£	128.70
CO-2000	Cola	Inventory shortage	£	98.28
CO-2000	Cola	Batch coding error	£	72.54
CO-600	Cola	Product spill	£	71.25
CO-600	Cola	Machine adjustment	£	71.25
CO-600	Cola	Other	£	56.25
CO-600	Cola	Batch coding error	£	55.00
CO-2000	Cola	Labeling error	£	51.48
CO-600	Cola	Label switch	£	25.00
CO-600	Cola	Batch change	£	25.00
CO-600	Cola	Conveyor belt jam	£	21.25
CO-2000	Cola	Other	£	16.38
CO-600	Cola	Calibration error	£	12.50
DC-600	Diet Cola	Machine failure	£	62.50
DC-600	Diet Cola	Inventory shortage	£	37.50
DC-600	Diet Cola	Batch coding error	£	25.00
DC-600	Diet Cola	Other	£	18.75
LE-600	Lemon lime	Batch change	£	100.00
LE-600	Lemon lime	Inventory shortage	£	31.25
LE-600	Lemon lime	Calibration error	£	30.00
LE-600	Lemon lime	Machine adjustment	£	25.00
LE-600	Lemon lime	Batch coding error	£	25.00
OR-600	Orange	Batch change	£	75.00
OR-600	Orange	Machine failure	£	18.75
RB-600	Root Berry	Machine adjustment	£	168.75
RB-600	Root Berry	Batch coding error	£	37.50
RB-600	Root Berry	Labeling error	£	25.00
RB-600	Root Berry	Inventory shortage	£	25.00
RB-600	Root Berry	Machine failure	£	22.50
RB-600	Root Berry	Calibration error	£	18.75
RB-600	Root Berry	Label switch	£	16.25
RB-600	Root Berry	Other	£	8.75

Strategic questions:

1. Efficiency and Productivity Improvement

a- How can we reduce downtime by 20-30% over the next five years?

By identifying the top 5 factors causing the most downtime, efforts can be focused on addressing and mitigating these key downtime causes, which can ultimately help reduce downtime by 20-30%.

```
SELECT df.description AS Downtime_cause
      ,SUM(ld.downtime_duration) AS total downtime
      ,count(*) AS occurrences
FROM line_downtime ld
JOIN downtime_factors df
ON ld.factor_id = df.id
GROUP BY df.description
HAVING count(*) >=5
ORDER BY total_downtime DESC
LIMIT 5;
```

The result

Downtime_cause	total_downtime	occurrences
Machine adjustment	332	12
Machine failure	254	11
Inventory shortage	225	9
Batch change	160	5
Batch coding error	145	6

b- What are the best global practices in downtime management that we can adopt?

The most 3 important practices are:

1. **Implement Predictive Maintenance (PdM)**

- **What it is: Predictive maintenance uses IoT sensors, machine learning, and historical data to predict when equipment is likely to fail or require maintenance. Unlike preventive maintenance, which is scheduled regularly, PdM focuses on real-time data to address potential issues before they become problems.**

- **Benefit: Reduces unexpected equipment failures, minimizes unplanned downtime, and optimizes maintenance schedules based on actual machine conditions.**

How to Adopt:

- **Invest in IoT sensors for key equipment.**
- **Use machine learning or AI software to analyze equipment data and predict failures.**
- **Train staff to respond quickly to predictive maintenance alerts.**

2. **Implement Downtime Tracking Systems**

- **automated downtime tracking software is used to monitor when, why, and for how long equipment is down.**
- **This data is critical for identifying patterns, bottlenecks, and areas for improvement.**

- **Benefit: Provides real-time data on downtime events, allowing for quick responses and long-term process improvements based on actionable insights.**

How to Adopt:

- **Install downtime tracking software that integrates with machines.**
- **Set up alerts for specific downtime events to notify relevant personnel immediately.**
- **Analyze downtime reports to identify recurring issues and address them proactively.**

3. **Improve Training and Cross-Training of Employees**

- **What it is: A well-trained workforce can minimize downtime by quickly responding to issues and performing necessary maintenance. Cross-training employees across different roles allows flexibility when key personnel are unavailable.**

- **Benefit: Ensures faster troubleshooting and response times, enhances the flexibility of operations, and reduces the risk of downtime due to skill gaps.**

<u>How to Adopt:</u>

- **Regularly update training programs for operators and technicians on new equipment and maintenance procedures.**
- **Cross-train employees to cover multiple roles to increase operational flexibility.**
- **Use simulations or hands-on training to improve workers' ability to respond to equipment failures.**

c- How can we design more flexible production processes to address future challenges?

```
SELECT lp.product_id AS Product,
       df.description AS downtime_cause,
       SUM(ld.downtime_duration) AS total_downtime
FROM line_downtime ld
JOIN line_productivity lp ON ld.batch = lp.batch
JOIN downtime_factors df ON ld.factor_id = df.id
GROUP BY lp.product_id, df.description
ORDER BY lp.product_id,total_downtime DESC;
```

product	downtime_cause	total_downtime
CO-2000	Machine adjustment	120
CO-2000	Machine failure	55
CO-2000	Inventory shortage	42
CO-2000	Batch coding error	31
CO-2000	Labeling error	22
CO-2000	Other	7
CO-600	Machine failure	116
CO-600	Inventory shortage	108
CO-600	Machine adjustment	57
CO-600	Product spill	57
CO-600	Other	45
CO-600	Batch coding error	44
CO-600	Batch change	20
CO-600	Label switch	20
CO-600	Conveyor belt jam	17
CO-600	Calibration error	10
DC-600	Machine failure	50
DC-600	Inventory shortage	30
DC-600	Batch coding error	20
DC-600	Other	15
LE-600	Batch change	80
LE-600	Inventory shortage	25
LE-600	Calibration error	24
LE-600	Batch coding error	20
LE-600	Machine adjustment	20
OR-600	Batch change	60
OR-600	Machine failure	15
RB-600	Machine adjustment	135
RB-600	Batch coding error	30
RB-600	Labeling error	20
RB-600	Inventory shortage	20
RB-600	Machine failure	18
RB-600	Calibration error	15
RB-600	Label switch	13
RB-600	Other	7

2.Digital Transformation and Technology

a- Can we use AI or machine learning to predict failures before they occur?

AI and machine learning can indeed be used to predict equipment failures before they occur, which falls under predictive maintenance. As a part of answering this question we have the most downtime causes that Occur frequently that can be predicted by AI and machine learning.

```
SELECT df.Description AS Downtime_cause
    ,COUNT(*) AS occurrences
    ,round(AVG(downtime_duration),2) AS avg_duration
FROM line_downtime ld
JOIN downtime_factors df ON ld.factor_id = df.id
GROUP BY 1
HAVING COUNT(*) > 5
ORDER BY occurrences DESC;
```

The result

downtime_cause	occurrences	avg_duration
Machine adjustment	12	27.67
Machine failure	11	23.09
Inventory shortage	9	25
Other	6	12.33
Batch coding error	6	24.17

b- How feasible is our investment in analytics systems like Power BI or ERP to accelerate downtime analysis?

- **Improved Real-Time Reporting and Visualization:**
- **Actionable Insights through AI/ML**
- **To support the feasibility analysis with SQL, we can analyze current data and downtime patterns to quantify how much downtime occurs, what the primary causes are, and the potential benefits of quicker insights through Power BI or ERP.**

Here's an example of how SQL queries can help justify the investment:

- **Downtime by Cause, #occurrences, and duration:**

```
SELECT
    df.Description AS Downtime_causes,
    COUNT(ld.Batch) AS occurrences,
    ROUND(AVG(ld.downtime_duration), 2) AS avg_duration,
    ROUND(SUM(ld.downtime_duration), 2) AS total_duration
FROM line_downtime ld
JOIN downtime_factors df ON ld.factor_id = df.id
GROUP BY df.Description
ORDER BY total_duration DESC
LIMIT 5;
```

The result

downtime_cause	occurrences	avg_duration	total_duration
Machine adjustment	12	27.67	332
Machine failure	11	23.09	254
Inventory shortage	9	25	225
Batch change	5	32	160
Batch coding error	6	24.17	145

3.Maintenance and Prevention Strategies

a- What is the optimal maintenance model: preventive or predictive?

To answer this question we needed to determine whether the downtime factors that occur the most are related to planned or scheduled maintenance OR due to unplanned events:

```
SELECT df.description
     ,SUM(ld.downtime_duration) AS total_downtime
     ,COUNT(ld.batch) AS occurrences
FROM line_downtime ld
JOIN downtime_factors df ON ld.factor_id = df.id
GROUP BY df.description
ORDER BY total_downtime DESC;
```

The result

downtime_cause	total_downtime	occurrences
Machine adjustment	332	12
Machine failure	254	11
Inventory shortage	225	9
Batch change	160	5
Batch coding error	145	6
Other	74	6
Product spill	57	3
Calibration error	49	3
Labeling error	42	2
Label switch	33	3
Conveyor belt jam	17	1

We noticed that the top 5 causes of downtime occur due to unplanned events, so the perfect model for our case would be predictive maintenance.

b-How can we reduce reliance on emergency maintenance caused by operators and increase planned maintenance?

Once we identify the factors with the highest number of operator error-related emergency occurrences, we can implement strategies to reduce them by offering Operator Training, Process Standardization, and Automation.

```
SELECT df.description
    ,COUNT(ld.factor_id) AS emergency_occurrences
FROM line_downtime ld
JOIN downtime_factors df ON ld.factor_id = df.id
WHERE df.operator_error = 'Yes'
GROUP BY df.description
ORDER BY emergency_occurrences DESC;
```

The result

downtime_cause	emergency_occurrences
Machine adjustment	12
Batch coding error	6
Batch change	5
Calibration error	3
Label switch	3
Product spill	3

c- How effective is our current maintenance schedule, and does it need redesigning based on data?

```
SELECT DATE_TRUNC('month', b_date) AS month
      ,SUM(downtime_duration) AS total_downtime
FROM line_downtime
JOIN line_productivity ON line_downtime.batch = line_productivity.batch
GROUP BY DATE_TRUNC('month', b_date)
ORDER BY total_downtime DESC
```

The result

month	total_downtime
Aug-24	853
Sep-24	535

```
SELECT
    CASE
        WHEN df.description LIKE '%maintenance%' THEN 'Maintenance'
        ELSE 'Other Factors'
    END AS downtime_type
   ,SUM(ld.downtime_duration) AS total_downtime
FROM line_downtime ld
JOIN downtime_factors df ON ld.factor_id = df.id
GROUP BY downtime_type
ORDER BY total_downtime DESC;
```

The result

downtime_type	total_downtime
Other Factors	1388
Maintenance	0

Note:

After running those SQL queries and listing the downtime factors in the dataset, we found that the company has no maintenance model, and it's necessary to design a predictive maintenance model.

4.Financial Impact and Profitability

a- How can we reduce financial losses from downtime by 50% in the coming years?

```
WITH DowntimeLosses AS (
    SELECT
        df.description
       ,SUM(ld.downtime_duration) AS total_downtime
       ,SUM(ld.downtime_duration * (pc.Unit_Price - pc.unit_cost)) AS total_loss
    FROM line_downtime ld
    JOIN downtime_factors df ON ld.factor_id = df.id
    JOIN line_productivity lp ON ld.Batch = lp.Batch
    JOIN products pc ON lp.product_id = pc.id
    GROUP BY df.description
)
SELECT
    description
   ,total_downtime
   ,total_loss
   ,round((total_loss - (total_loss * 50 /100)) ,2) AS loss_after_reduction
FROM DowntimeLosses
ORDER BY total_loss DESC;
```

The result

downtime_cause	total_downtime	total_loss	loss_after_reduction
Machine adjustment	332	£ 109.16	£ 54.58
Machine failure	254	£ 75.49	£ 37.75
Inventory shortage	225	£ 65.41	£ 32.70
Batch coding error	145	£ 43.01	£ 21.50
Batch change	160	£ 40.00	£ 20.00
Other	74	£ 20.03	£ 10.01
Labeling error	42	£ 15.30	£ 7.65
Product spill	57	£ 14.25	£ 7.13
Calibration error	49	£ 12.25	£ 6.13
Label switch	33	£ 8.25	£ 4.13
Conveyor belt jam	17	£ 4.25	£ 2.13

b- Are there alternative ways to compensate for downtime losses?

One way to compensate for downtime is to increase production efficiency during operational hours. This means optimizing the production process to produce more output in less time, offsetting the lost output during downtime.

```
SELECT
        operator
   ,EXTRACT(HOUR FROM start_time) AS shift
   ,round(AVG(no_of_bottles)) AS avg_output_per_shift
FROM line_productivity
GROUP BY 1, 2
ORDER BY avg_output_per_shift DESC;
```

operator	shift	avg_output_per_shift
Mac	11	27000
Dee	5	24600
Mac	12	22000
Mac	15	22000
Dee	4	21500
Dee	1	21000
Dennis	12	20900
Charlie	17	20800
Mac	14	20500
Dennis	10	20300
Mac	17	20000
Dennis	14	20000
Dee	7	19200
Charlie	19	18650
Dee	9	18000
Dennis	8	18000
Charlie	20	17200
Dee	6	17000
Charlie	18	16600
Dee	11	16000
Dee	2	16000
Charlie	16	16000
Charlie	15	16000
Dennis	9	15000
Charlie	21	15000
Charlie	22	15000
Dennis	7	13400
Mac	22	13000
Charlie	14	12000
Mac	16	12000
Dee	10	12000

5.Strategic Planning and Expansion

a- How will downtimes affect our ability to scale production?

1. Reduced Production Capacity:

Downtimes directly reduce the time available for production, limiting the total output. As a result, when scaling production (increasing output to meet higher demand), frequent downtimes can prevent it from reaching full capacity, which slows down overall growth.

```
SELECT
    EXTRACT(MONTH FROM b_date) AS month
   ,SUM(downtime_duration) AS total_downtime
   ,(SUM(total_batch_time) - SUM(downtime_duration)) AS
available_production_time
FROM line_downtime ld
JOIN line_productivity lp ON ld.batch = lp.batch
GROUP BY month
ORDER BY month;
```

month	total_downtime	available_production_time	total_production
Aug-24	853	3104	791400
Sep-24	535	2363	401700

b- Can we redesign production processes to be more resilient to downtimes?

Here are some ways to redesign production processes:

1. **Optimizing Maintenance Schedules**
2. **Automation and Predictive Maintenance**
3. **Improved Communication and Real-Time Monitoring.**

Python syntax(matplotlib)for creating control charts

1.Control Chart for Batch Time Efficiency

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

# Create dataframe from line_productivity
data = {
    'batch': [422111, 422112, 422113, 422114, 422115, 422116, 422117, 422118, 
422119, 422120,
              422121, 422122, 422123, 422124, 422125, 422126, 422127, 422128, 
422129, 422130,
              422131, 422132, 422133, 422134, 422135, 422136, 422137, 422138, 
422139, 422140,
              422141, 422142, 422143, 422144, 422145, 422146, 422147, 422148],
    'product_id': ['OR-600', 'LE-600', 'LE-600', 'LE-600', 'LE-600', 'LE-600', 'LE-
600', 'CO-600',
                   'CO-600', 'CO-600', 'CO-600', 'CO-600', 'CO-600', 'CO-600', 'CO-600', 
'CO-600',
                   'CO-600', 'CO-600', 'CO-600', 'CO-600', 'CO-600', 'CO-600', 'DC-600', 
'DC-600',
                   'DC-600', 'DC-600', 'RB-600', 'RB-600', 'RB-600', 'RB-600', 'RB-600', 
'RB-600',
                   'RB-600', 'CO-2000', 'CO-2000', 'CO-2000', 'CO-2000', 'CO-2000'],
    'total_batch_time': [135, 100, 110, 100, 84, 60, 75, 120, 85, 112, 75, 85, 133, 
100, 80, 104,
                         83, 112, 75, 80, 90, 60, 80, 110, 105, 60, 105, 80, 95, 123, 67, 90, 
118,
                         152, 120, 160, 205, 130],
    'Actual Batch time': [60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 
60, 60, 60,
```

```
            60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 60, 98, 98, 98,
            98, 98]
}

df = pd. DataFrame(data)
# Calculate efficiency ratio (Actual/Expected)
df['time_efficiency'] = df['Actual Batch time'] / df['total_batch_time']

# Calculate control limits
mean_efficiency = df['time_efficiency'].mean()
std_efficiency = df['time_efficiency'].std()
ucl = mean_efficiency + 3*std_efficiency
lcl = mean_efficiency - 3*std_efficiency

# Plot control chart
plt.figure(figsize=(12, 6))
plt.plot(df['batch'], df['time_efficiency'], 'bo-', label='Time Efficiency')
plt.axhline(mean_efficiency, color='g', linestyle='--', label='Mean')
plt.axhline(ucl, color='r', linestyle='--', label='UCL')
plt.axhline(lcl, color='r', linestyle='--', label='LCL')
plt.title('Control Chart for Batch Time Efficiency (Actual/Total Time)')
plt.xlabel('Batch Number')
plt.ylabel('Efficiency Ratio')
plt.xticks(rotation=45)
plt.legend()
plt.grid(False)
plt.tight_layout()
plt.show()
```

2. Control Chart for Production Output Efficiency

```
# Create dataframe with production data
production_data = {
    'batch': [422111, 422112, 422113, 422114, 422115, 422116, 422117, 422118, 422119, 422120,
              422121, 422122, 422123, 422124, 422125, 422126, 422127, 422128, 422129, 422130,
              422131, 422132, 422133, 422134, 422135, 422136, 422137, 422138, 422139, 422140,
              422141, 422142, 422143, 422144, 422145, 422146, 422147, 422148],
    'Expected no_of_bottles': [27000, 20000, 22000, 20000, 16800, 12000, 15000, 24000, 17000, 22400,
                               15000, 17000, 26600, 20000, 16000, 20800, 16600, 22400, 15000, 16000,
                               18000, 12000, 16000, 22000, 21000, 12000, 21000, 16000, 19000, 24600,
                               13400, 18000, 23600, 15200, 12000, 16000, 20500, 13000],
    'Actual # Bottels': [12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000,
                         12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000,
                         12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000, 12000,
                         12000, 12000, 12000, 9800, 9800, 9800, 9800, 9800]
}

prod_df = pd.DataFrame(production_data)
```

```python
# Calculate production efficiency
prod_df['production_efficiency'] = prod_df['Actual # Bottels'] /
prod_df['Expected no_of_bottles']

# Calculate control limits
mean_prod_eff = prod_df['production_efficiency'].mean()
std_prod_eff = prod_df['production_efficiency'].std()
ucl_prod = mean_prod_eff + 3*std_prod_eff
lcl_prod = mean_prod_eff - 3*std_prod_eff

# Plot control chart
plt.figure(figsize=(12, 6))
plt.plot(prod_df['batch'], prod_df['production_efficiency'], 'go-',
label='Production Efficiency')
plt.axhline(mean_prod_eff, color='b', linestyle='--', label='Mean')
plt.axhline(ucl_prod, color='r', linestyle='--', label='UCL')
plt.axhline(lcl_prod, color='r', linestyle='--', label='LCL')
plt.title('Control Chart for Production Efficiency (Actual/Expected Bottles)')
plt.xlabel('Batch Number')
plt.ylabel('Efficiency Ratio')
plt.xticks(rotation=45)
plt.legend()
plt.grid(False)
plt.tight_layout()
plt.show()
```

3. Downtime Analysis by Operator

```
# Create merged dataframe with downtime and operator info
downtime_data = {
    'batch': [422111, 422111, 422112, 422112, 422113, 422114, 422114, 422115,
422117, 422117,
             422118, 422118, 422118, 422118, 422119, 422120, 422120, 422120,
422121, 422122,
             422123, 422123, 422124, 422124, 422125, 422125, 422126, 422127,
422128, 422128,
             422129, 422130, 422131, 422131, 422133, 422134, 422134, 422135,
422135, 422137,
             422137, 422138, 422139, 422139, 422140, 422140, 422141, 422142,
422143, 422143,
             422144, 422144, 422145, 422146, 422146, 422146, 422147, 422147,
422147, 422148,
             422148],
    'factor_id': [2, 7, 2, 8, 2, 4, 6, 10, 2, 6, 6, 7, 11, 12, 4, 4, 5, 9, 7, 7, 4, 7, 5, 6,
                 11, 12, 8, 6, 5, 7, 12, 2, 4, 10, 7, 7, 8, 4, 12, 8, 10, 3, 4, 6, 6, 11, 12,
                 6, 6, 7, 6, 8, 3, 6, 7, 12, 4, 6, 7, 4, 8],
    'downtime_duration': [60, 15, 20, 20, 50, 25, 15, 24, 10, 5, 14, 16, 10, 20, 25,
20, 15, 17,
                         15, 25, 43, 30, 20, 20, 10, 10, 44, 23, 22, 30, 15, 20, 20, 10, 20, 30,
                         20, 30, 15, 30, 15, 20, 20, 15, 50, 13, 7, 30, 40, 18, 30, 24, 22, 30,
                         25, 7, 17, 60, 30, 25, 7]
}

downtime_df = pd.DataFrame(downtime_data)
operators = {
```

```
    'batch': [422111, 422112, 422113, 422114, 422115, 422116, 422117, 422118,
422119, 422120,
              422121, 422122, 422123, 422124, 422125, 422126, 422127, 422128,
422129, 422130,
              422131, 422132, 422133, 422134, 422135, 422136, 422137, 422138,
422139, 422140,
              422141, 422142, 422143, 422144, 422145, 422146, 422147, 422148],
    'operator': ['Mac', 'Mac', 'Mac', 'Mac', 'Charlie', 'Charlie', 'Charlie', 'Dee',
'Dee', 'Dee',
                 'Dennis', 'Dennis', 'Dennis', 'Dennis', 'Charlie', 'Charlie', 'Charlie',
'Charlie',
                 'Charlie', 'Dee', 'Dee', 'Dee', 'Dee', 'Mac', 'Mac', 'Mac', 'Dee', 'Dee',
'Dee',
                 'Dee', 'Dennis', 'Dennis', 'Dennis', 'Dennis', 'Charlie', 'Charlie',
'Charlie', 'Mac']
}

operator_df = pd.DataFrame(operators)

# Merge data
merged_df = pd.merge(downtime_df, operator_df, on='batch')

# Get operator error info from downtime_factors
operator_error_factors = [2, 5, 6, 8, 10, 11]  # From downtime_factors sheet
where operator_error=True

# Classify downtime
merged_df['is_operator_error'] =
merged_df['factor_id'].isin(operator_error_factors)
```

```
# Group by operator and error type
downtime_summary = merged_df.groupby(['operator',
'is_operator_error'])['downtime_duration'].sum().unstack()
downtime_summary.columns = ['Non-Operator Error', 'Operator Error']

# Plot stacked bar chart
downtime_summary.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.title('Downtime Minutes by Operator and Error Type')
plt.xlabel('Operator')
plt.ylabel('Total Downtime Minutes')
plt.xticks(rotation=0)
plt.grid(False)
plt.tight_layout()
plt.show()
```