Below is a proposed software specification that defines a set of Python API functions. Each function is designed to address one or more of the instructions in the domain of semiconductor chip test data analytics. You can adjust the parameters as needed when you implement the actual code.

**Title of the function**: Function 1 – Validate and Visualize Summary  
**Function Name**: validate\_and\_visualize\_summary  
**ID**: 36\_6  
**Purpose**: Validate the summary of E‑test measurements per wafer for accuracy and generate simple visualizations (e.g., bar charts) that depict the counts per wafer.  
**Signature**:

def validate\_and\_visualize\_summary(summary\_data: pd.DataFrame, visualize: bool = True) -> pd.DataFrame:

"""

Validates the summary data of E-test measurements per wafer for accuracy.

Optionally generates visualizations (e.g., bar charts) to depict the counts per wafer.

Parameters:

summary\_data (pd.DataFrame): DataFrame containing the wafer summary.

visualize (bool): Whether to generate visualizations.

Returns:

pd.DataFrame: The validated summary data.

"""

**Used For**:

* Workflow 36\_6 – “Summarize the number of E-test measurements per wafer. Validate and Visualize: Optionally, review the summary for accuracy and create simple visualizations (e.g., bar charts) to depict the counts per wafer.”

**Title of the function**: Function 2 – Process Chart (Validation, Annotation, and Saving)  
**Function Name**: process\_chart  
**ID**: 74\_6, 36\_6  
**Purpose**: Validate a chart for accuracy and either annotate it (adding labels or clarifications) or save it for presentation/further analysis.  
**Signature**:

def process\_chart(chart\_data: Any, annotate: bool = False, annotations: dict = None,

save: bool = False, save\_path: str = None) -> Any:

"""

Validates the provided chart for accuracy.

Optionally annotates the chart with provided labels and saves it if requested.

Parameters:

chart\_data (Any): The chart object or data to be processed.

annotate (bool): Flag to indicate if the chart should be annotated.

annotations (dict): Dictionary of annotations/labels to add to the chart.

save (bool): Flag to indicate if the chart should be saved.

save\_path (str): File path where the chart should be saved.

Returns:

Any: The processed chart (possibly annotated and/or saved).

"""

**Used For**:

* Workflow 74\_6 – “Generate a simple yield trend chart for the past three weeks. Review and Save Chart: Validate the trend chart for accuracy and prepare it for presentation or further analysis.”
* Workflow 36\_6 – “Plot the trend of lot-level yield over the last 6 weeks. Validate and Annotate Chart: Review the plotted chart for accuracy and add any necessary annotations or labels for clarity.”

**Title of the function**: Function 3 – Export Lot Yield Table  
**Function Name**: export\_lot\_yield\_table  
**ID**: 64\_5, 64\_4  
**Purpose**: Validate the computed table of lot-level yields for accuracy and format it with lot identifiers and metadata for export or display as a report.  
**Signature**:

def export\_lot\_yield\_table(lot\_data: pd.DataFrame, report\_format: str = "table") -> pd.DataFrame:

"""

Validates and formats the lot-level yield data.

Exports or displays the data as a report in the specified format.

Parameters:

lot\_data (pd.DataFrame): DataFrame containing lot-level yields and related metadata.

report\_format (str): The format in which to export/display the table (e.g., "table", "csv").

Returns:

pd.DataFrame: The validated and formatted table.

"""

**Used For**:

* Workflow 64\_5 – “Provide a table of lot-level yields for the current quarter. Validate and Export Table - Cross-check the table for accuracy and export or display the result as a report.”
* Workflow 64\_4 – “Organize Data into a Table - Format the computed lot-level yields along with lot identifiers and relevant metadata into a table.”

**Title of the function**: Function 4 – Recommend Process Change  
**Function Name**: recommend\_process\_change  
**ID**: 58\_7  
**Purpose**: Analyze wafer edge E‑test data and failing die locations to determine if a process change is needed, then recommend adjusting manufacturing parameters at the wafer edge.  
**Signature**:

def recommend\_process\_change(edge\_data: pd.DataFrame, correlation\_threshold: float = 0.8) -> dict:

"""

Analyzes edge-site E-test data and failing die locations.

If a strong correlation is found, recommends a process change for manufacturing parameters at the wafer edge.

Parameters:

edge\_data (pd.DataFrame): DataFrame containing edge-site E-test metrics and die failure data.

correlation\_threshold (float): Threshold to decide if the correlation is strong.

Returns:

dict: A dictionary with recommendations and supporting analysis.

"""

**Used For**:

* Workflow 58\_7 – “Determine if project DEF789’s low yield is due to a shift in E-test metrics at the wafer edge; correlate edge site E-test values to failing die locations and recommend whether a process change is needed. Recommend Process Change.”

**Title of the function**: Function 5 – Recommend Corrective Actions  
**Function Name**: recommend\_corrective\_actions  
**ID**: 68\_7  
**Purpose**: Analyze spatial distributions of failing dies and identified correlations/root causes to suggest targeted corrective actions (e.g., process re‑optimization, equipment recalibration, foundry adjustments).  
**Signature**:

def recommend\_corrective\_actions(correlation\_data: pd.DataFrame, analysis\_params: dict = None) -> dict:

"""

Evaluates the spatial distribution of failing dies and correlates them with manufacturing data.

Suggests corrective actions based on identified correlations and root causes.

Parameters:

correlation\_data (pd.DataFrame): DataFrame containing spatial and failure data.

analysis\_params (dict): Additional parameters for analysis (optional).

Returns:

dict: Recommendations for corrective actions.

"""

**Used For**:

* Workflow 68\_7 – “Correlate the spatial distribution of failing dies with specific manufacturing process adjustments and suggest possible corrective actions.”

**Title of the function**: Function 6 – Recommend PCM Adjustments  
**Function Name**: recommend\_pcm\_adjustments  
**ID**: 13\_8  
**Purpose**: Analyze whether specific PCM parameters predict low FT1 yield and recommend adjustments to PCM test locations or foundry tuning knobs.  
**Signature**:

def recommend\_pcm\_adjustments(pcm\_data: pd.DataFrame, yield\_threshold: float = 0.9) -> dict:

"""

Analyzes PCM parameters in relation to FT1 yield.

Recommends adjustments to PCM test locations or foundry tuning knobs to mitigate low yield.

Parameters:

pcm\_data (pd.DataFrame): DataFrame containing PCM test measurements and yield data.

yield\_threshold (float): The yield threshold used to determine low performance.

Returns:

dict: A set of recommended adjustments.

"""

**Used For**:

* Workflow 13\_8 – “Detect if certain PCM parameters predict low FT1 yield, isolate the wafers where this correlation is highest, and recommend changes to either PCM test locations or foundry tuning knobs. Recommendation Formulation.”

**Title of the function**: Function 7 – Suggest Retest Strategy  
**Function Name**: suggest\_retest\_strategy  
**ID**: 15\_6  
**Purpose**: Based on analysis of wafers with yields below 90%, determine if retesting might improve yield and outline potential retest strategies targeting the identified issues.  
**Signature**:

def suggest\_retest\_strategy(test\_data: pd.DataFrame, current\_yield: float) -> dict:

"""

Analyzes test measurements for wafers with low yields.

Recommends a retest strategy if analysis indicates potential yield improvement.

Parameters:

test\_data (pd.DataFrame): DataFrame containing test measurements.

current\_yield (float): The current yield value to compare against.

Returns:

dict: A recommendation detailing retest strategies.

"""

**Used For**:

* Workflow 15\_6 – “For all wafers in the past 2 months with yields below 90%, ... Suggest Retest Strategy - Based on the analysis, make a recommendation on whether retesting could potentially improve yield, and if so, outline potential retest strategies targeting the identified issues.”

**Title of the function**: Function 8 – Output Failing Test Measurements  
**Function Name**: output\_failing\_measurements  
**ID**: 11\_5  
**Purpose**: Present a list of test measurements that resulted in failing outcomes, including details such as failure type and measurement values.  
**Signature**:

def output\_failing\_measurements(measurements: pd.DataFrame) -> None:

"""

Presents the list of failing test measurements with additional details (e.g., failure type, measurement values).

Parameters:

measurements (pd.DataFrame): DataFrame containing test measurement results.

Returns:

None

"""

**Used For**:

* Workflow 11\_5 – “List test measurements that resulted in failing outcomes for a selected wafer. Output Results.”

**Title of the function**: Function 9 – Output Top Failure Bins  
**Function Name**: output\_top\_failure\_bins  
**ID**: 52\_5  
**Purpose**: Format and list the top three test bins with the highest failure counts for easy review.  
**Signature**:

def output\_top\_failure\_bins(bin\_data: pd.DataFrame, top\_n: int = 3) -> None:

"""

Formats and outputs the top failure bins along with their corresponding failure counts.

Parameters:

bin\_data (pd.DataFrame): DataFrame containing test bin failure counts.

top\_n (int): The number of top bins to output.

Returns:

None

"""

**Used For**:

* Workflow 52\_5 – “List the top three test bins with the highest failure counts. Output Results.”

**Title of the function**: Function 10 – Display Average Parametric Value  
**Function Name**: display\_average\_value  
**ID**: 80\_6  
**Purpose**: Present the computed average value of a selected parametric test measurement clearly, using either a report display or dashboard visualization.  
**Signature**:

def display\_average\_value(average\_value: float, display\_format: str = "dashboard") -> None:

"""

Presents the computed average parametric test value in a clear format.

Parameters:

average\_value (float): The computed average value.

display\_format (str): The format for displaying the value (e.g., "dashboard", "report").

Returns:

None

"""

**Used For**:

* Workflow 80\_6 – “Display the average parametric test value for a chosen test measurement. Display the Result.”

**Title of the function**: Function 11 – Adjust Histogram Parameters  
**Function Name**: adjust\_histogram\_parameters  
**ID**: 29\_6  
**Purpose**: Fine-tune the histogram visualization by setting appropriate bins, axis labels, and titles to improve clarity.  
**Signature**:

def adjust\_histogram\_parameters(histogram\_data: Any, bins: int, axis\_labels: dict, title: str) -> Any:

"""

Fine-tunes a histogram visualization by adjusting parameters.

Parameters:

histogram\_data (Any): The data or chart object representing the histogram.

bins (int): Number of bins for the histogram.

axis\_labels (dict): Dictionary with axis label definitions (e.g., {"x": "Measurement", "y": "Count"}).

title (str): Title for the histogram.

Returns:

Any: The updated histogram visualization.

"""

**Used For**:

* Workflow 29\_6 – “Generate a histogram of a selected E-test measurement across wafers in the latest lot. Adjust Histogram Parameters.”

**Title of the function**: Function 12 – Generate Comprehensive Report  
**Function Name**: generate\_report  
**ID**: 6\_9, 34\_10, 62\_8, 2\_9, 8\_8, 31\_9, 24\_9, 7\_7, 11\_7  
**Purpose**: Consolidate analysis results, visualizations, insights, and recommendations into a detailed and comprehensive report.  
**Signature**:

def generate\_report(analysis\_results: dict, visualizations: list = None,

recommendations: dict = None, report\_title: str = "Analysis Report") -> str:

"""

Generates a comprehensive report that consolidates analysis findings, visualizations, and recommendations.

Parameters:

analysis\_results (dict): A dictionary containing key analysis metrics and insights.

visualizations (list): A list of visualization objects or paths to images (optional).

recommendations (dict): A dictionary with recommendation details (optional).

report\_title (str): The title of the report.

Returns:

str: The formatted report as a string.

"""

**Used For**:

* Workflow 6\_9 – “Document Findings - Create a report summarizing the methodology, results, visualizations, and any potential recommendations for process adjustments.”
* Workflow 34\_10 – “Document Findings - Consolidate the analysis, insights, and visualizations into a comprehensive report.” (used for both lots tested on tester T001 and temporal correlation analyses)
* Workflow 62\_8 – “Interpret Findings and Report Results: Analyze the statistical and visual outputs ... compile the analysis into a final report.”
* Workflow 2\_9 – “Interpretation and Reporting - Consolidate the analysis and interpret whether specific test programs ... compile a detailed report.”
* Workflow 8\_8 – “Interpret and Report Findings - Summarize the analysis ... and include recommendations.”
* Workflow 31\_9 – “Interpret and Document Findings - Summarize whether one tester consistently underperforms ...”
* Workflow 24\_9 – “Interpret and Report Findings: Summarize the analysis, including statistical significance, trends, and recommendations.”
* Workflow 7\_7 – “Interpretation and Reporting: Summarize the findings in a report highlighting significant variation ...”
* Workflow 11\_7 – “Visualization and Reporting – Create visual reports and charts comparing original and simulated yields ...”

**Title of the function**: Function 13 – Create Visual Plots  
**Function Name**: create\_visual\_plots  
**ID**: 1\_10  
**Purpose**: Create a variety of visual plots—such as wafer maps with highlighted pattern regions, correlation scatter plots, and bar charts—to illustrate key findings.  
**Signature**:

def create\_visual\_plots(data: pd.DataFrame, plot\_types: list, title: str = "Visual Plots") -> Any:

"""

Creates visual plots from the provided data.

Parameters:

data (pd.DataFrame): The data to be visualized.

plot\_types (list): A list specifying the types of plots to generate (e.g., ["wafer\_map", "scatter", "bar"]).

title (str): The title for the set of plots.

Returns:

Any: The generated visual plots.

"""

**Used For**:

* Workflow 1\_10 – “Examine all lots from Test House X with yield drops ... Visualization – Create visual plots (like wafer maps, correlation scatter plots, and bar charts) to illustrate the findings.”

**Title of the function**: Function 14 – Create Visualizations  
**Function Name**: create\_visualizations  
**ID**: 3\_9  
**Purpose**: Generate time series plots and box plots to support the analysis of trends (e.g., E-test fluctuations, parametric measurement variability) for selected lots.  
**Signature**:

def create\_visualizations(data: pd.DataFrame, plot\_options: dict) -> Any:

"""

Generates visualizations such as time series and box plots to support analysis.

Parameters:

data (pd.DataFrame): Data used for creating visualizations.

plot\_options (dict): Options for plot customization (e.g., plot type, axis labels).

Returns:

Any: The generated visualization objects.

"""

**Used For**:

* Workflow 3\_9 – “Monitor recent lots with yield between 92% and 95%, detect trends in E-test fluctuations, ... Visualization – Create visualizations (e.g., time series plots for E-test trends, box plots for parametric measurement variability).”

**Title of the function**: Function 15 – Aggregate Failure Rates  
**Function Name**: aggregate\_failure\_rates  
**ID**: 43\_5  
**Purpose**: Organize and aggregate computed failure rates into a table or dataset to enable easy comparison across different test programs.  
**Signature**:

def aggregate\_failure\_rates(failure\_data: pd.DataFrame, group\_by: str = "test\_program") -> pd.DataFrame:

"""

Aggregates failure rates by the specified grouping (e.g., test program).

Parameters:

failure\_data (pd.DataFrame): DataFrame containing failure rate data.

group\_by (str): The field to group the data by (default is "test\_program").

Returns:

pd.DataFrame: A DataFrame with aggregated failure rates.

"""

**Used For**:

* Workflow 43\_5 – “Compare logical test failure rates across different test programs. Aggregate Results - Organize the computed failure rates in a table or dataset for easy comparison.”

**Title of the function**: Function 16 – Highlight Identified Wafers  
**Function Name**: highlight\_wafers  
**ID**: 35\_4, 23\_5  
**Purpose**: Generate a summary (table and/or visualization) that clearly highlights wafers with yields below a specified threshold (e.g., below 95% or below 90%).  
**Signature**:

def highlight\_wafers(wafer\_data: pd.DataFrame, yield\_threshold: float) -> pd.DataFrame:

"""

Highlights wafers whose yield is below the specified threshold.

Parameters:

wafer\_data (pd.DataFrame): DataFrame containing wafer yield data.

yield\_threshold (float): The yield threshold (e.g., 95 or 90) below which wafers are highlighted.

Returns:

pd.DataFrame: A DataFrame summarizing or marking the identified wafers.

"""

**Used For**:

* Workflow 35\_4 – “Highlight any wafers with yield below the target of 95%. Highlight Identified Wafers.”
* Workflow 23\_5 – “Highlight any wafers with yield below 90% in the latest lot. Highlight Identified Wafers.”

This specification outlines 16 functions covering validation, visualization, recommendation, data export, output, aggregation, and reporting. Each API function’s signature includes a brief docstring that explains its parameters and purpose, providing a clear roadmap for implementation.