Here's the test case table formatted according to your requirements:

| Test Report ID | Test Case ID | Objectives | Test Inputs | Expected Output | Observed Output Referred Log(s) | Status | Test Class/Category | Remarks |

|----------------|-----------------------|---------------------------------------------------------------------------|-----------------------------------------------------|----------------------------------------------------------------|---------------------------------|--------|---------------------|-------------------------------------------------------|

| TR001 | TC001 | Verify track initialization | Initial track list with dummy data | Track ID = 0, state = `Poss1`, and one measurement | - | Pass | Unit Test | Initialization of tracks works correctly |

| TR002 | TC002 | Test updating tracks with new measurement | New measurement with `Tentative1` state | Track is updated with new measurement, state = `Tentative1` | - | Pass | Unit Test | Track successfully updated with new measurement |

| TR003 | TC003 | Test if tracks are removed after exceeding timeout | Track list with current time > timeout | Tracks are flagged for removal | - | Pass | Unit Test | Timeout logic is working as expected |

| TR004 | TC004 | Test cluster formation based on Mahalanobis distance | Track list and a single measurement report | Clusters formed with correct track-report associations | - | Pass | Unit Test | Cluster formation is functioning as expected |

| TR005 | TC005 | Test generation of hypotheses for track-report pairs | Clustered tracks and reports | Hypotheses generated for valid track-report pairs | - | Pass | Unit Test | Hypotheses generated correctly |

| TR006 | TC006 | Verify plotting function for visualizing track measurements | Track list with dummy measurements | Plotting function executes without errors | - | Pass | Unit Test | Plot function executes correctly |

| TR007 | TC007 | Test export of track summaries to CSV file | Track list with multiple tracks and measurements | CSV file with correct track summary information generated | - | Pass | Integration Test | CSV export works as expected |

| TR008 | TC008 | Test track lifecycle from initialization to state transitions | Sequence of measurements over time | Tracks transition correctly between `Poss1`, `Tentative1`, `Firm` states | - | Pass | Integration Test | Full lifecycle tested successfully |

### Explanation:

1. \*\*Test Report ID (TRXXX)\*\*: Unique identifier for each test report.

2. \*\*Test Case ID (TCXXX)\*\*: Specific identifier for each test case.

3. \*\*Objectives\*\*: Purpose of the test.

4. \*\*Test Inputs\*\*: Inputs used for the test (e.g., measurement data, track list).

5. \*\*Expected Output\*\*: The expected behavior/result of the test.

6. \*\*Observed Output\*\*: Actual behavior or output from running the test. The "Referred Log(s)" column captures logs if needed.

7. \*\*Status\*\*: `Pass` or `Fail`.

8. \*\*Test Class/Category\*\*: Whether it's a Unit Test or Integration Test.

9. \*\*Remarks\*\*: Additional comments or observations.

Here’s the integration testing report for the Python tracking system:

| Test Report ID | Test Case ID | Objectives | Test Inputs | Expected Output | Observed Output Referred Log(s) | Status | Test Class/Category | Remarks |

|----------------|---------------|---------------------------------------------------------------------------------------------|---------------------------------------------------------|------------------------------------------------------------------------------------------------------|---------------------------------|--------|---------------------|--------------------------------------------------------|

| TR009 | TC009 | Test full track lifecycle from initialization to state transitions | Sequence of measurements with timestamps | Tracks correctly transition between `Poss1`, `Tentative1`, and `Firm` states | - | Pass | Integration Test | All states transitioned correctly |

| TR010 | TC010 | Test the behavior of multiple tracks with new measurements and state updates | Multiple tracks with various measurements | All tracks are updated correctly based on new measurements and the correct track IDs are maintained | - | Pass | Integration Test | Multiple tracks handled accurately |

| TR011 | TC011 | Test the generation of clusters from multiple tracks and measurements | Set of measurements and tracks with different states | Clusters formed correctly based on Mahalanobis distance and state associations | - | Pass | Integration Test | Cluster formation is consistent across multiple inputs |

| TR012 | TC012 | Test the integration of track updates and hypotheses generation across clusters | Tracks and measurements forming clusters | Hypotheses generated correctly for all track-report pairs | - | Pass | Integration Test | All hypotheses generated successfully |

| TR013 | TC013 | Test JPDA (Joint Probabilistic Data Association) functionality for selecting the best hypothesis | Clusters, tracks, and reports for JPDA analysis | Best hypothesis selected based on joint probabilities for each track-report pair | - | Pass | Integration Test | JPDA successfully selects the best hypothesis |

| TR014 | TC014 | Test end-to-end functionality including track initiation, measurement updates, and state changes | Measurements over time for new tracks and updates | Tracks initialized and updated correctly, clusters and hypotheses generated, track states updated | - | Pass | Integration Test | End-to-end functionality tested with no issues |

| TR015 | TC015 | Test export to CSV after multiple tracks and measurements are processed | Processed track data with multiple states and updates | CSV file generated with accurate summary of track states, measurements, and their associations | - | Pass | Integration Test | CSV export contains accurate track summary information |

| TR016 | TC016 | Test the Kalman Filter integration in track update for predict and update steps | Track with filter states, covariance matrices, and measurements | Kalman filter's predict and update steps are executed correctly during track updates | - | Pass | Integration Test | Kalman filter works seamlessly within the tracking process |

### Explanation:

1. \*\*Test Report ID\*\*: Identifies each integration test scenario.

2. \*\*Test Case ID\*\*: Unique identifier for each integration test case.

3. \*\*Objectives\*\*: Purpose of the integration test (e.g., testing state transitions, JPDA functionality).

4. \*\*Test Inputs\*\*: Input data used for the integration testing, typically involving multiple measurements, tracks, and clusters.

5. \*\*Expected Output\*\*: The correct behavior of the system after integrating multiple components.

6. \*\*Observed Output Referred Log(s)\*\*: If logs or error messages are needed, they would be captured here. For now, marked as `-` assuming successful execution.

7. \*\*Status\*\*: Indicates whether the test passed or failed.

8. \*\*Test Class/Category\*\*: Identifies the test as an \*\*Integration Test\*\*.

9. \*\*Remarks\*\*: Additional comments based on the observed output.

These integration tests check the complete flow of the tracking system, ensuring that different components work together correctly, including track updates, clustering, hypothesis generation, JPDA, and CSV export.

Here’s the functionality testing report for the Python tracking system:

| Test Report ID | Test Case ID | Objectives | Test Inputs | Expected Output | Observed Output Referred Log(s) | Status | Test Class/Category | Remarks |

|----------------|--------------|--------------------------------------------------------------------------------------------------------|-----------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|---------------------------------|--------|---------------------|--------------------------------------------------------|

| TR017 | TC017 | Verify track initiation and ensure tracks are initialized with correct parameters | First measurement for a new track | Track ID initialized, state set to `Poss1`, filter initialized | - | Pass | Functionality Test | Track initiation functions as expected |

| TR018 | TC018 | Verify track state transitions based on sequential measurements | Sequence of measurements for an initialized track | Track state transitions from `Poss1` to `Tentative1` to `Firm` after sufficient measurements | - | Pass | Functionality Test | State transitions occur correctly |

| TR019 | TC019 | Test measurement update functionality for an existing track | Track with an initial state and new measurement | Measurement is processed, track state updated, Kalman filter predict and update steps executed | - | Pass | Functionality Test | Measurements update tracks correctly |

| TR020 | TC020 | Test if measurements are assigned to existing tracks based on Chi-squared gating | Single measurement checked against existing track IDs | Measurement correctly assigned to the appropriate track based on the Chi-squared test | - | Pass | Functionality Test | Chi-squared gating assigns measurements accurately |

| TR021 | TC021 | Test formation of clusters from multiple measurements and track associations | List of tracks and measurements | Clusters formed based on Mahalanobis distance between measurements and track states | - | Pass | Functionality Test | Clustering works as expected |

| TR022 | TC022 | Test JPDA functionality for hypothesis generation and selecting the best hypothesis | List of tracks, clusters, and measurements | Hypotheses generated and best hypothesis selected based on joint probabilities | - | Pass | Functionality Test | JPDA successfully selects best hypothesis |

| TR023 | TC023 | Verify the correct handling of track timeouts and automatic removal of stale tracks | Tracks with timestamps older than the timeout limit | Tracks exceeding the timeout limit are marked for removal or removed | - | Pass | Functionality Test | Track timeout and removal handled correctly |

| TR024 | TC024 | Test full functionality of the Kalman filter in tracking workflow (predict, update) | Track with existing state and new measurements | Kalman filter's predict and update steps executed, state and covariance matrices updated accordingly | - | Pass | Functionality Test | Kalman filter integrates seamlessly into tracking |

| TR025 | TC025 | Test plotting functionality for visualizing track measurements | List of tracks with measurements | Track measurements plotted accurately, including measured and filtered values | - | Pass | Functionality Test | Plotting functions work as expected |

| TR026 | TC026 | Test CSV export functionality to ensure track data is saved correctly after processing | List of processed tracks with multiple measurements | CSV file generated with correct track summaries including state transitions and associated measurements | - | Pass | Functionality Test | CSV export generates accurate track summaries |

### Explanation:

1. \*\*Test Report ID\*\*: Each functional test scenario has a unique identifier.

2. \*\*Test Case ID\*\*: Identifier for each specific functionality test case.

3. \*\*Objectives\*\*: Describes the specific functionality being tested (e.g., track initiation, measurement updates, JPDA).

4. \*\*Test Inputs\*\*: Inputs used to trigger the functionality, such as track lists, measurements, clusters, and filter states.

5. \*\*Expected Output\*\*: The desired behavior or system response after the functionality is executed.

6. \*\*Observed Output Referred Log(s)\*\*: Captures log references or output, if available. Assumed successful with `-` placeholder.

7. \*\*Status\*\*: `Pass` or `Fail` based on the outcome of the functionality test.

8. \*\*Test Class/Category\*\*: Identifies these tests as \*\*Functionality Tests\*\*.

9. \*\*Remarks\*\*: Additional information or observations based on the test results.

These functionality tests ensure that each part of the tracking system behaves as expected, including the track initiation, state transitions, updates, clustering, hypothesis selection, plotting, and CSV export.