Creating QueryPairs using the Query Wizard

Creating QueryPairs using the Query Wizard in QuerySurge involves leveraging its graphical interface to build pairs of SQL queries that can be executed sequentially to validate and compare data. QueryPairs in QuerySurge are used for data testing, validation, and verification across different datasets or environments. Here’s how you can create QueryPairs using the Query Wizard in QuerySurge:

**Steps to Create QueryPairs in QuerySurge:**

1. **Access QuerySurge**:
   * Log in to your QuerySurge environment and navigate to the Query Wizard feature. This typically involves accessing a web-based interface provided by QuerySurge.
2. **Start Query Wizard**:
   * Initiate the Query Wizard tool. This tool helps streamline the creation of SQL queries and their organization into pairs for testing purposes.
3. **Define QueryPairs**:
   * Within the Query Wizard, specify the type of QueryPairs you want to create. Common types include:
     + **Baseline and Test QueryPairs**: These pairs are used to compare results between a baseline dataset (expected results) and a test dataset (actual results).
     + **Source and Target QueryPairs**: Used to compare data between two different data sources or tables to ensure data integrity during ETL (Extract, Transform, Load) processes.
     + **Before and After QueryPairs**: Used to validate changes or updates made to data, comparing datasets before and after a specific operation.
4. **Select Data Sources**:
   * Choose the relevant data sources or databases from which you want to retrieve data for your queries. QuerySurge allows you to connect to various types of databases and data platforms.
5. **Build SQL Queries**:
   * Use the graphical interface provided by QuerySurge to construct SQL queries for each part of the QueryPair. This interface often includes tools for drag-and-drop query building, syntax highlighting, and query validation.
6. **Parameterization (if needed)**:
   * Define parameters within your SQL queries if you need to make them dynamic. Parameters allow you to reuse the same queries with different inputs, enhancing flexibility and scalability.
7. **Configure Execution Settings**:
   * Configure any settings related to query execution, such as timeout durations, result formatting, or logging options. Ensure that these settings align with your testing requirements and environment constraints.
8. **Test QueryPairs**:
   * Before deploying QueryPairs in a production or testing environment, run them within QuerySurge to validate their functionality and accuracy. Verify that the queries retrieve and compare data as expected.
9. **Save QueryPairs**:
   * Once satisfied with the QueryPairs, save them within QuerySurge. Provide meaningful names and descriptions to facilitate easy identification and reuse in future testing cycles.
10. **Document and Share**:
    * Document the purpose and scope of each QueryPair. This documentation helps in understanding the testing approach and sharing knowledge with team members involved in data validation and testing.
11. **Execute QueryPairs**:
    * Execute the QueryPairs as part of your testing processes. Monitor the execution results within QuerySurge to identify any discrepancies or issues between datasets as per your validation criteria.
12. **Review and Analyze Results**:
    * Analyze the results generated by QuerySurge after executing the QueryPairs. Review any discrepancies found and investigate their root causes if necessary.

By following these steps, you can effectively create QueryPairs using the Query Wizard in QuerySurge, leveraging its capabilities to automate and streamline data validation and testing processes across different datasets or environments.

Top of Form

Bottom of Form

Creating a TestSuite and executing it

In QuerySurge, creating a TestSuite involves organizing a collection of test cases that validate the data and ensure the integrity of your ETL (Extract, Transform, Load) processes or data migrations. A TestSuite allows you to group related tests together for easier management and execution. Here’s a step-by-step guide on how to create a TestSuite and execute it in QuerySurge:

**Creating a TestSuite in QuerySurge**

1. **Access QuerySurge**:
   * Log in to your QuerySurge environment using your credentials.
2. **Navigate to Test Management**:
   * Once logged in, navigate to the Test Management section within QuerySurge. This is where you can create and manage your TestSuites and test cases.
3. **Create a New TestSuite**:
   * Click on "New TestSuite" or a similar option to create a new TestSuite. Give your TestSuite a descriptive name that reflects its purpose or the type of tests it will contain.
4. **Add Test Cases**:
   * Within the TestSuite interface, add individual test cases that you want to include in this TestSuite. Test cases can be created based on SQL queries, file comparisons, API tests, or any other validations supported by QuerySurge.
5. **Organize Test Cases**:
   * Arrange the test cases in the order you want them to be executed within the TestSuite. This organization is important if you have dependencies between tests or specific sequencing requirements.
6. **Configure TestSuite Settings**:
   * Configure any settings specific to the TestSuite, such as:
     + **Execution Parameters**: Specify parameters that apply to all tests within the TestSuite, such as database connections or environment variables.
     + **Execution Order**: Define the order in which tests should be executed.
     + **Timeouts and Constraints**: Set timeouts or constraints that apply to the entire TestSuite execution.
7. **Save the TestSuite**:
   * Once you’ve added and configured your test cases, save the TestSuite within QuerySurge. This step ensures that your TestSuite is stored and can be easily accessed for future executions or modifications.

**Executing a TestSuite in QuerySurge**

1. **Select the TestSuite**:
   * Navigate to the Test Management section of QuerySurge and locate the TestSuite you want to execute. Click on the TestSuite to open its details.
2. **Configure Execution Options**:
   * Before executing the TestSuite, configure any execution options specific to this run:
     + **Environment Selection**: Choose the environment (such as development, testing, or production) against which the TestSuite should be executed.
     + **Execution Mode**: Decide whether to run the tests immediately or schedule them for a later time.
3. **Execute the TestSuite**:
   * Initiate the execution of the TestSuite. Depending on your selection, QuerySurge will begin running each test case sequentially or in parallel, based on your configuration.
4. **Monitor Execution Progress**:
   * Monitor the progress of the TestSuite execution within QuerySurge. The interface typically provides real-time updates on test execution status, including pass/fail results for each test case.
5. **Review Results**:
   * After the TestSuite execution completes, review the results generated by QuerySurge. Identify any test cases that failed and investigate the reasons for failure.
6. **Generate Reports**:
   * Generate detailed reports summarizing the outcomes of the TestSuite execution. These reports provide insights into the quality and reliability of your data processes or migrations.
7. **Document Findings**:
   * Document any issues found during testing, along with steps taken to resolve them. This documentation helps in tracking the progress of testing and ensuring that all identified issues are addressed.
8. **Iterate and Improve**:
   * Use the insights gained from TestSuite execution to iterate on your tests and improve their effectiveness. Update TestSuites as needed based on changes to your data environments or business requirements.

By following these steps, you can effectively create a TestSuite and execute it in QuerySurge, ensuring thorough validation and verification of your data processes and ETL operations.

Top of Form

Bottom of Form

* Creating a Table-level comparison using the Query Wizard

Creating a table-level comparison using the Query Wizard in QuerySurge involves setting up queries that compare data between two or more tables within your data sources. This process is essential for verifying data consistency, validating ETL processes, or ensuring accurate data migrations. Here’s a step-by-step guide on how to create a table-level comparison using the Query Wizard in QuerySurge:

**Steps to Create a Table-level Comparison in QuerySurge**

1. **Access QuerySurge**:
   * Log in to your QuerySurge environment using your credentials.
2. **Navigate to Query Wizards**:
   * In QuerySurge, locate and access the Query Wizards feature. This is typically found under a section related to creating and managing queries.
3. **Start a New Query Wizard Session**:
   * Initiate a new Query Wizard session to begin building your table-level comparison queries. Look for options like "New Query" or "Create QueryPair" depending on how QuerySurge labels its query creation process.
4. **Select Data Sources**:
   * Choose the data sources (tables or databases) that you want to compare. QuerySurge allows you to connect to various data sources such as databases (SQL Server, Oracle, MySQL, etc.), flat files, and more.
5. **Define Comparison Criteria**:
   * Specify how you want to compare the tables. Common comparison criteria include:
     + **Column-wise Comparison**: Compare specific columns across tables to identify matching or differing values.
     + **Row-wise Comparison**: Compare entire rows to find exact matches or discrepancies.
     + **Aggregated Comparison**: Compare aggregate functions (like sum, count, average) across tables to assess overall data consistency.
6. **Write SQL Queries**:
   * Use the graphical interface provided by QuerySurge to construct SQL queries for your comparison. You can typically drag-and-drop columns, use query builders, or directly write SQL statements depending on the capabilities of QuerySurge’s Query Wizard.
7. **Implement Join Conditions (if necessary)**:
   * If you are comparing tables that are related through keys (e.g., primary keys, foreign keys), ensure you implement appropriate join conditions in your queries. This allows you to match corresponding rows accurately.
8. **Include Comparison Logic**:
   * Incorporate logic into your queries to perform the comparison. Use SQL operators and functions to identify matching records or discrepancies between tables based on your chosen comparison criteria.
9. **Test Queries**:
   * Before finalizing, test each query within QuerySurge to ensure it retrieves and compares data as expected. Verify that the logic correctly identifies matches, differences, or inconsistencies.
10. **Save the Comparison Query**:
    * Once satisfied with your queries, save them within QuerySurge. Give them meaningful names and descriptions to facilitate easy identification and reuse in future comparisons.
11. **Execute the Comparison Query**:
    * Execute the saved comparison queries to perform the table-level comparison. Monitor the execution within QuerySurge and review the results generated.
12. **Review and Analyze Results**:
    * Analyze the results of the table-level comparison to identify any discrepancies or differences between the tables being compared. QuerySurge typically provides detailed reports and insights into the comparison results.
13. **Document Findings**:
    * Document the findings of your table-level comparison. Note any discrepancies found, potential causes, and recommended actions for resolution or further investigation.

By following these steps, you can effectively create a table-level comparison using the Query Wizard in QuerySurge, leveraging SQL queries to ensure data integrity and accuracy across your databases or data sources. This process is crucial for maintaining quality assurance in data-driven projects and ensuring that data processes meet expected standards.

Top of Form

Bottom of Form