File Edit View Insert Cell Kernel Widgets Help Not Trusted | Kernel

Run Code

Edit Your Expectation Suite

Use this notebook to recreate and modify your expectation suite:

Expectation Suite Name: `test_results`

We'd love it if you **reach out to us on** the [Great Expectations Slack Channel]

In [1]:
```
import datetime
import great_expectations as ge
import great_expectations.jupyter_ux
from great_expectations.checkpoint import LegacyCheckpoint
from great_expectations.data_context.types.resource_identifiers import ValidationResultIdentifier

context = ge.data_context.DataContext()

# Feel free to change the name of your suite here. Renaming this will not
# remove the other one.
expectation_suite_name = "test_results"
suite = context.get_expectation_suite(expectation_suite_name)
suite.expectations = []

batch_kwargs = {'data_asset_name': 'results', 'datasource': 'great_expect', 'limit': 1000, 'schema': 'wsdc_results', 'table': 're
batch = context.get_batch(batch_kwargs, suite)
batch.head()
```

```
2021-06-28T01:36:55+0530 - INFO - Great Expectations logging enabled at 20 level by JupyterUX module.
2021-06-28T01:36:56+0530 - INFO - Generating query from table batch_kwargs based on limit and offset
2021-06-28T01:36:56+0530 - INFO -       0 expectation(s) included in expectation_suite.
```

Out[1]:

	reference_id	Msno	BatchCode	SecName	DegCode	YrSemCode	SlNo	RegNo	Name	Grade1	Grade2	Grade3	Grade4	Grade5	Grade6	Grade7	G
0	1	0	2011	None	1	41	None	8148	M. Sunny Davidson	B	B	C	None	B	None	A	
1	1	0	2011	None	1	41	None	9133	Durgam Praveen	None	None	None	None	None	None	None	
2	1	0	2011	None	1	41	None	9155	Khaja Mohammed Abdul Qavi	B	D	None	None	None	D	None	
3	1	0	2011	None	1	41	None	101113	Bhagavatula Jitendra Sai	P	None	None	None	None	F	None	
4	1	0	2011	None	1	41	None	101135	Mohammad Ali	D	P	None	None	None	None	P	

Create & Edit Expectations

Add expectations by calling specific expectation methods on the `batch` object. They all begin with `.expect_` which makes autocompleting easy using tab.

You can see all the available expectations in the **expectation glossary**.

Table Expectation(s)

In [2]:
```
batch.expect_table_row_count_to_be_between(max_value=1100, min_value=900)
```

Out[2]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 1000
  }
}
```

In [3]:
```
batch.expect_table_column_count_to_equal(value=29)
```

Out[3]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 29
  }
}
```

In [4]:
```
batch.expect_table_columns_to_match_ordered_list(column_list=['reference_id', 'Msno', 'BatchCode', 'SecName', 'DegCode', 'YrSemCo
```

Out[4]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": [
      "reference_id",
      "Msno",
      "BatchCode",
      "SecName",
      "DegCode",
      "YrSemCode",
      "SlNo",
      "RegNo",
      "Name",
      "Grade1",
      "Grade2",
      "Grade3",
      "Grade4",
      "Grade5",
      "Grade6",
      "Grade7",
      "Grade8",
      "Grade9",
      "Grade10",
      "Grade11",
      "Grade12",
      "Grade13",
      "Grade14",
      "Sgpa",
      "Cgpa",
      "TotCr1",
      "TotCr2",
      "Nof1",
      "Nof2"
    ]
  }
}
```

Column Expectation(s)

RegNo

In [5]:
```
batch.expect_column_to_exist(column='RegNo')
```

Out[5]:

```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {}
}
```


In [6]:
```
batch.expect_column_values_to_not_be_null(column='RegNo')
```

Out[6]:

```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 1000,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "partial_unexpected_list": []
  }
}
```


In [7]:
```
batch.expect_column_values_to_be_unique(column='RegNo')
```

Out[7]:

```
{
  "success": false,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 1000,
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_count": 86,
    "unexpected_percent": 8.6,
    "unexpected_percent_total": 8.6,
    "unexpected_percent_nonmissing": 8.6,
    "partial_unexpected_list": [
      9155,
      101113,
      101238,
      111122,
      111139,
      111206,
      111217,
      111221,
      9155,
      101113,
      101171,
      101201,
      101238,
      111108,
      111117,
      111122,
      111139,
      111143,
      111157,
      111206
    ]
  }
}
```


BatchCode

In [8]:
```
batch.expect_column_to_exist(column='BatchCode')
```

Out[8]:

```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {}
}
```


Nithin

In [9]:
```
batch.expect_column_to_exist(column='Nithin')
```

Out[9]:

```
{
  "success": false,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {}
}
```


reference_id

In [10]:
```
batch.expect_column_values_to_not_be_null(column='reference_id')
```

Out[10]:

```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 1000,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "partial_unexpected_list": []
  }
}
```


In [11]:
```
batch.expect_column_distinct_values_to_be_in_set(column='reference_id', value_set=[1, 2, 4, 5, 6, 7, 8, 27])
```

Out[11]:

```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": [
      1,
      2,
      4,
```

```
      5,
      6,
      7,
      8,
      27
    ],
    "element_count": 1000,
    "missing_count": null,
    "missing_percent": null
  }
}
```

In [12]:
```
batch.expect_column_kl_divergence_to_be_less_than(column='reference_id', partition_object={'values': [1, 2, 4, 5, 6, 7, 8, 27],
```

Out[12]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 0.0,
    "element_count": 1000,
    "missing_count": null,
    "missing_percent": null
  }
}
```

`SlNo`

In [13]:
```
batch.expect_column_values_to_not_be_null(column='SlNo', mostly=0.841)
```

Out[13]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 1000,
    "unexpected_count": 59,
    "unexpected_percent": 5.899999999999995,
    "unexpected_percent_total": 5.899999999999995,
    "partial_unexpected_list": []
  }
}
```

In [14]:
```
batch.expect_column_min_to_be_between(column='SlNo', max_value=2, min_value=0)
```

Out[14]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 1,
    "element_count": 1000,
    "missing_count": 59,
    "missing_percent": 5.9
  }
}
```

In [15]:
```
batch.expect_column_max_to_be_between(column='SlNo', max_value=1000, min_value=998)
```

Out[15]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 999,
    "element_count": 1000,
    "missing_count": 59,
    "missing_percent": 5.9
  }
}
```

In [16]:
```
batch.expect_column_mean_to_be_between(column='SlNo', max_value=159.1286, min_value=157.1286)
```

Out[16]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 158.1286,
    "element_count": 1000,
    "missing_count": 59,
    "missing_percent": 5.9
  }
}
```

In [17]:
```
batch.expect_column_median_to_be_between(column='SlNo', max_value=64, min_value=62)
```

Out[17]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "observed_value": 63,
    "element_count": 1000,
    "missing_count": 59,
    "missing_percent": 5.9
  }
}
```

`Name`

In [18]:
```
batch.expect_column_values_to_not_be_null(column='Name')
```

Out[18]:
```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 1000,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "partial_unexpected_list": []
  }
}
```

In [19]:
```
batch.expect_column_value_lengths_to_be_between(column='Name', min_value=1)
```

Out[19]:

```
{
  "success": true,
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  },
  "meta": {},
  "result": {
    "element_count": 1000,
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "unexpected_percent_nonmissing": 0.0,
    "partial_unexpected_list": []
  }
}
```


Save & Review Your Expectations

Let's save the expectation suite as a JSON file in the `great_expectations/expectations` directory of your project. If you decide not to save some expectations that you created, use remove_expectation method.

Let's now rebuild your Data Docs, which helps you communicate about your data with both machines and humans.

In [20]:
```
batch.save_expectation_suite(discard_failed_expectations=False)

results = LegacyCheckpoint(
    name="_temp_checkpoint",
    data_context=context,
    batches=[
        {
          "batch_kwargs": batch_kwargs,
          "expectation_suite_names": [expectation_suite_name]
        }
    ]
).run()
validation_result_identifier = results.list_validation_result_identifiers()[0]
context.build_data_docs()
context.open_data_docs(validation_result_identifier)
```

```
2021-06-28T01:36:57+0530 - INFO -       18 expectation(s) included in expectation_suite. result_format settings filtered.
2021-06-28T01:36:57+0530 - INFO - Generating query from table batch_kwargs based on limit and offset
2021-06-28T01:36:57+0530 - INFO - Setting run_name to: 20210627T200657.638606Z
2021-06-28T01:36:57+0530 - INFO -       18 expectation(s) included in expectation_suite.
```