

Add status Add categories

Run all Graph

Detecting Financial Stress Periods in Students

This project analyzes student attendance and simulated fee payment delays to detect periods of financial stress. By combining attendance patterns with delay-based stress classification, the dashboard identifies students who may be at academic risk due to financial difficulties.

+ Add project filter

SQL 1

4h ago {} Dataframes

```

1 select
2   Student_ID,
3   count(*) as Total_Classes,
4   sum(Attendance) as Classes_Attended,
5   round( (sum(Attendance) * 100.0) / count(*), 2 ) as Attendance_Rate,
6   date '2025-01-10' as Fee_Due_Date,
7   date '2025-01-10' + ((CAST(Student_ID AS INTEGER) % 30) * interval '1 day') as Fee_Paid_Date,
8
9   -- delay days
10  date_diff(
11    'day',
12    date '2025-01-10',
13    CAST(
14      date '2025-01-10' + ((CAST(Student_ID AS INTEGER) % 30) * interval '1 day')
15      AS DATE
16    )
17  ) as Delay_Days,
18
19  -- stress level based on delay
20  case
21    when date_diff(
22      'day',
23      date '2025-01-10',
24      CAST(
25        date '2025-01-10' + ((CAST(Student_ID AS INTEGER) % 30) * interval '1 day')
26        AS DATE
27      )
28    ) <= 5 then 'Low Stress'
29
30    when date_diff(
31      'day',
32      date '2025-01-10',
33      CAST(
34        date '2025-01-10' + ((CAST(Student_ID AS INTEGER) % 30) * interval '1 day')
35        AS DATE
36      )
37    ) between 6 and 15 then 'Medium Stress'
38
39    else 'High Stress'
40  end as Stress_Level
41
42 from "attendance_clean.csv"
43 group by Student_ID;
44
45
46
47
48

```

Filters

	# Student_ID	# Total_Classes	# Classes_Attended	# Attendance_Rate	Fee_Due_Date	Fee_Paid_Date	# Delay_Days	A Stress_Level	+
0	201.0	1	1.0	100.0	2025-01-10	2025-01-31T00:00:00	21	High Stress	
1	202.0	1	0.0	0.0	2025-01-10	2025-02-01T00:00:00	22	High Stress	
2	210.0	1	0.0	0.0	2025-01-10	2025-01-10T00:00:00	0	Low Stress	
3	217.0	1	1.0	100.0	2025-01-10	2025-01-17T00:00:00	7	Medium Stress	
4	218.0	1	0.0	0.0	2025-01-10	2025-01-18T00:00:00	8	Medium Stress	
5	220.0	1	1.0	100.0	2025-01-10	2025-01-20T00:00:00	10	Medium Stress	
6	222.0	1	1.0	100.0	2025-01-10	2025-01-22T00:00:00	12	Medium Stress	
7	223.0	1	1.0	100.0	2025-01-10	2025-01-23T00:00:00	13	Medium Stress	
8	227.0	1	1.0	100.0	2025-01-10	2025-01-27T00:00:00	17	High Stress	
9	228.0	1	0.0	0.0	2025-01-10	2025-01-28T00:00:00	18	High Stress	
10	229.0	1	1.0	100.0	2025-01-10	2025-01-29T00:00:00	19	High Stress	
11	231.0	1	1.0	100.0	2025-01-10	2025-01-31T00:00:00	21	High Stress	
12	232.0	1	0.0	0.0	2025-01-10	2025-02-01T00:00:00	22	High Stress	
13	237.0	1	1.0	100.0	2025-01-10	2025-02-06T00:00:00	27	High Stress	
14	240.0	1	1.0	100.0	2025-01-10	2025-01-10T00:00:00	0	Low Stress	
15	246.0	1	1.0	100.0	2025-01-10	2025-01-16T00:00:00	6	Medium Stress	

↳ attendance_clean

< 1 2 ... 393 > View Format Columns Totals 19,602 rows ↴

SQL 2

```

1 select
2   Stress_Level,
3   count(*) as Student_Count
4 from (
5   -- your final query that already creates Stress_Level
6   select
7     Student_ID,
8     case
9       when date_diff(
10         'day',
11         date '2025-01-10',
12         CAST(
13           date '2025-01-10' + ((CAST(Student_ID AS INTEGER) % 30) * interval '1 day')
14           AS DATE
15         )
16       ) <= 5 then 'Low Stress'
17     when date_diff(
18       'day',
19       date '2025-01-10',
20       CAST(
21         date '2025-01-10' + ((CAST(Student_ID AS INTEGER) % 30) * interval '1 day')
22         AS DATE
23       )
24     ) between 6 and 15 then 'Medium Stress'
25     else 'High Stress'
26   end as Stress_Level
27   from "attendance_clean.csv"
28   group by Student_ID
29 )
30 group by Stress_Level;
31

```

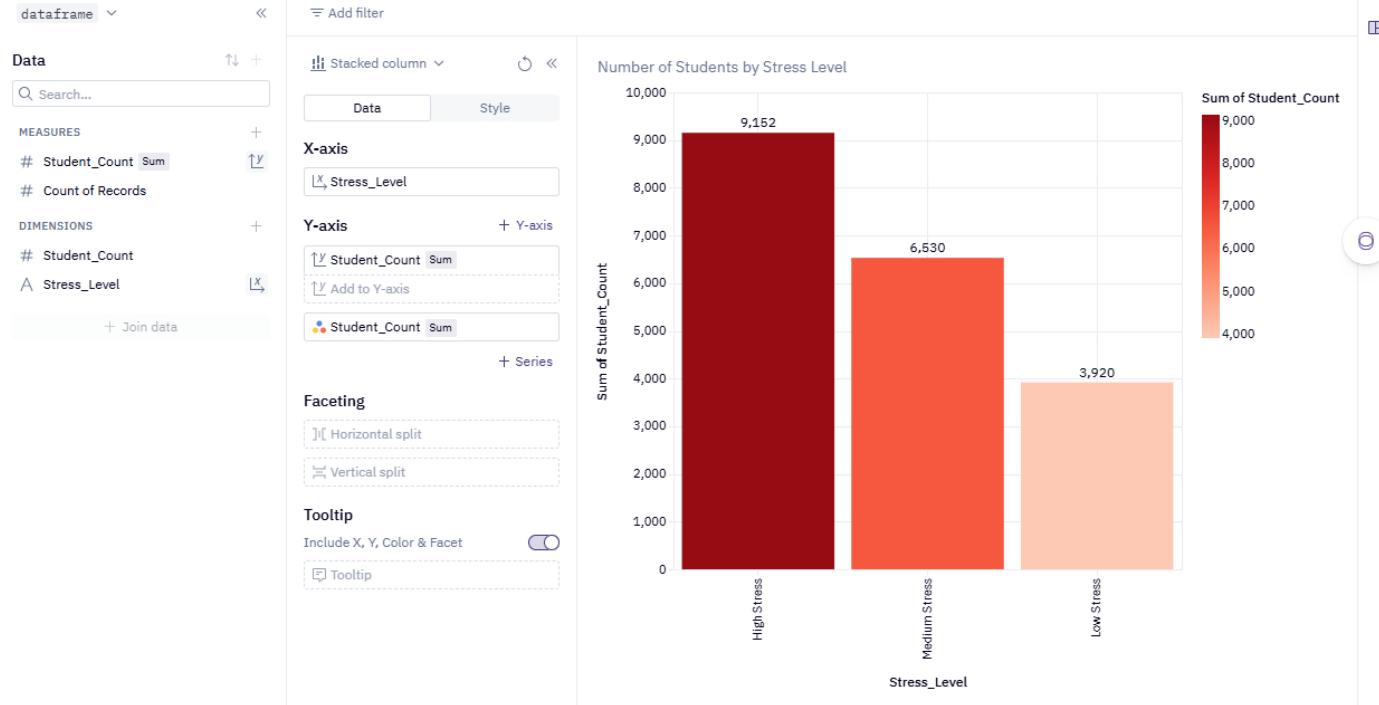
Filters

	A Stress_Level	# Student_Count	+
0	Low Stress	3920	
1	High Stress	9152	
2	Medium Stress	6530	

↳ dataframe 1

3 rows ↴

Number of Students by Stress Level



SQL 4

```

1 select
2   Stress_Level,
3   round(avg(Attendance_Rate), 2) as Avg_Attendance_Rate
4 from (
5   -- your main query that already creates Stress_Level and Attendance_Rate
6   select
7     Student_ID,
8     round((sum(Attendance) * 100.0) / count(*), 2) as Attendance_Rate,
9     case

```

```

10    when date_diff(
11        'day',
12        date '2025-01-10',
13        cast(
14            | date '2025-01-10' + ((cast(Student_ID as integer) % 30) * interval '1 day')
15            | as date)
16    ) <= 5 then 'Low Stress'
17
18    when date_diff(
19        'day',
20        date '2025-01-10',
21        cast(
22            | date '2025-01-10' + ((cast(Student_ID as integer) % 30) * interval '1 day')
23            | as date)
24    ) between 6 and 15 then 'Medium Stress'
25
26    else 'High Stress'
27 end as Stress_Level
28 from "attendance_clean.csv"
29 group by Student_ID
30 ) t
31 group by Stress_Level;
32

```

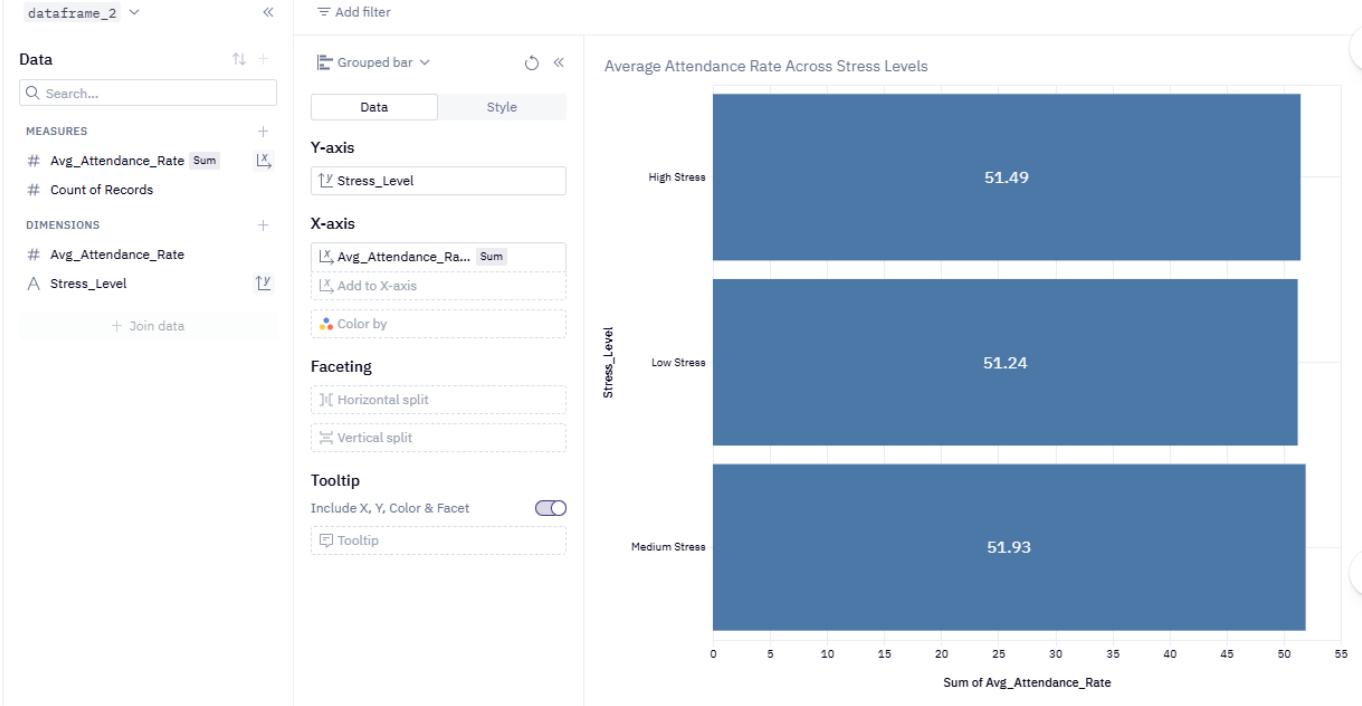
Filters

	A Stress_Level	# Avg_Attendance_Rate	+
0	Low Stress	51.24	
1	High Stress	51.49	
2	Medium Stress	51.93	

↳ `dataframe_2` 1

3 rows ↴

Average Attendance Rate Across Stress Levels



SQL 6

```

1 select
2     Delay_Days,
3     round(avg(Attendance_Rate), 2) as Avg_Attendance_Rate
4 from (
5     select
6         Student_ID,
7         round((sum(Attendance) * 100.0) / count(*), 2) as Attendance_Rate,
8
9         date_diff(
10            'day',
11            date '2025-01-10',
12            cast(
13                | date '2025-01-10' + ((cast(Student_ID as integer) % 30) * interval '1 day')
14                | as date)
15        ) as Delay_Days
16
17     from "attendance_clean.csv"
18     group by Student_ID
19 ) t
20 group by Delay_Days

```

```
21 order by Delay_Days;
22
```

Filters

	# Delay_Days	# Avg_Attendance_Rate	+
0	0	52.6	
1	1	52.83	
2	2	52.68	
3	3	48.85	
4	4	49.31	
5	5	51.15	
6	6	51.91	
7	7	47.17	
8	8	54.52	
9	9	52.99	
10	10	51.91	
11	11	50.54	
12	12	56.2	
13	13	51.91	
14	14	51.61	
15	15	50.54	

↳ **dataframe_3** 1

30 rows ↴



Key Takeaway

Students with longer fee payment delays are more likely to experience high financial stress and show slightly lower and irregular attendance patterns. Early identification of delayed payments can help institutions prevent academic risk.

Insights & Conclusion

1. A significant proportion of students are classified under High Stress, indicating that delayed fee payments are a frequent and serious concern in the student population.
2. Students with High Stress levels show a slightly lower average attendance rate compared to those in Low Stress, suggesting that financial difficulties may negatively influence classroom participation and academic engagement.
3. The analysis shows a clear relationship between payment delay and stress level. As the number of delay days increases, the likelihood of a student falling into Medium or High Stress categories rises steadily.
4. The trend between Delay Days and Attendance Rate indicates that prolonged financial stress leads to more irregular attendance patterns, which may affect long-term academic performance.

Final Conclusion:

Fee payment delay can be used as a strong early indicator of student financial stress and potential academic risk.

By combining fee delay data with attendance patterns, educational institutions can proactively identify vulnerable students and provide timely financial support, counseling, or flexible payment plans.

This data-driven approach can improve student well-being, reduce drop-out risk, and enhance overall academic success.

The screenshot displays a user interface for data analysis and visualization. At the top, there is a horizontal toolbar with several icons and labels: Agent, SQL Query, Python, Text (with a dropdown arrow), Chart, Pivot, Single value, Table, Inputs, Data, and More. Below the toolbar, the main workspace shows a Python 3.11 (Medium) notebook containing 9 cells. To the right of the notebook, there is a run history section with three entries: "2 hrs ago", "4 hrs ago", and "2 hrs ago". Further to the right, it shows the "Run mode: Auto" setting and a red button labeled "Stopped". In the bottom right corner, there is a circular profile icon with a purple "A".