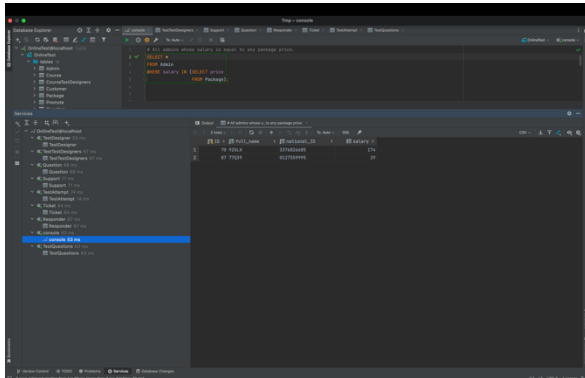
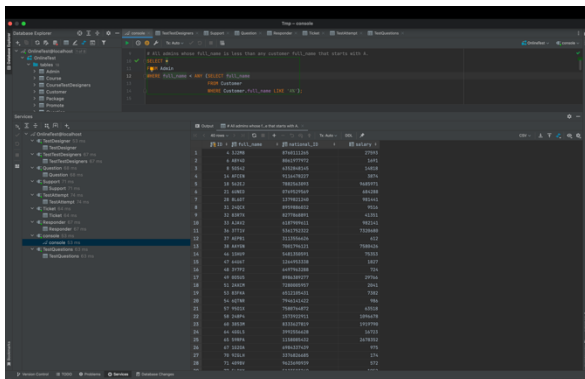


1. AdminIn:



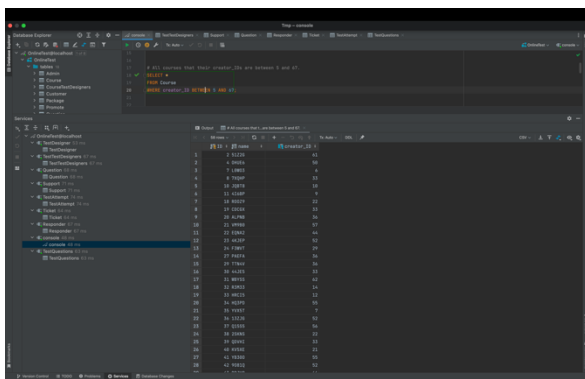
All admins whose salary is equal to any package price.

2. AdminAny:



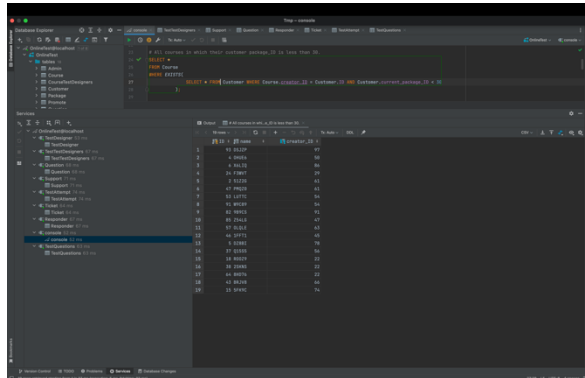
All admins whose full_name is less than any customer full_name that starts with A.

3. CourseBetween:



All courses that their creator_IDs are between 5 and 67.

4. CourseExists:

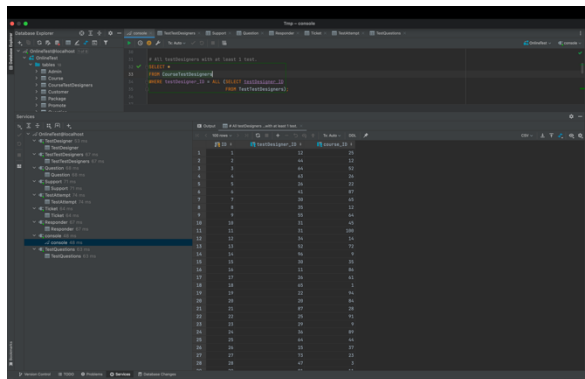


The screenshot shows a Tableau workspace with a calculated field named 'CourseExists' defined as: `IF (COUNT([CustomerPackageID]) > 0) THEN 'Yes' ELSE 'No' END`. Below the definition is a table of data with columns: Course, CustomerPackageID, and CourseExists. The table lists various courses and their corresponding customer package IDs, with 'Yes' or 'No' indicating the existence of the course.

Course	CustomerPackageID	CourseExists
1	1	Yes
2	2	Yes
3	3	Yes
4	4	Yes
5	5	Yes
6	6	Yes
7	7	Yes
8	8	Yes
9	9	Yes
10	10	Yes
11	11	Yes
12	12	Yes
13	13	Yes
14	14	Yes
15	15	Yes
16	16	Yes
17	17	Yes
18	18	Yes
19	19	Yes
20	20	Yes
21	21	Yes
22	22	Yes
23	23	Yes
24	24	Yes
25	25	Yes
26	26	Yes
27	27	Yes
28	28	Yes
29	29	Yes
30	30	Yes
31	31	Yes
32	32	Yes
33	33	Yes
34	34	Yes
35	35	Yes
36	36	Yes
37	37	Yes
38	38	Yes
39	39	Yes
40	40	Yes
41	41	Yes
42	42	Yes
43	43	Yes
44	44	Yes
45	45	Yes
46	46	Yes
47	47	Yes
48	48	Yes
49	49	Yes
50	50	Yes
51	51	Yes
52	52	Yes
53	53	Yes
54	54	Yes
55	55	Yes
56	56	Yes
57	57	Yes
58	58	Yes
59	59	Yes
60	60	Yes
61	61	Yes
62	62	Yes
63	63	Yes
64	64	Yes
65	65	Yes
66	66	Yes
67	67	Yes
68	68	Yes
69	69	Yes
70	70	Yes
71	71	Yes
72	72	Yes
73	73	Yes
74	74	Yes
75	75	Yes
76	76	Yes
77	77	Yes
78	78	Yes
79	79	Yes
80	80	Yes
81	81	Yes
82	82	Yes
83	83	Yes
84	84	Yes
85	85	Yes
86	86	Yes
87	87	Yes
88	88	Yes
89	89	Yes
90	90	Yes
91	91	Yes
92	92	Yes
93	93	Yes
94	94	Yes
95	95	Yes
96	96	Yes
97	97	Yes
98	98	Yes
99	99	Yes
100	100	Yes

All courses in which their customer package_ID is less than 30.

5. CourseTestDesignersAll:

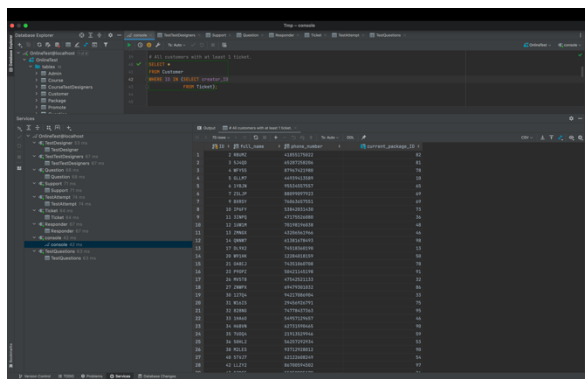


The screenshot shows a Tableau workspace with a calculated field named 'CourseTestDesignersAll' defined as: `IF (COUNT([TestDesignerID]) > 0) THEN 'Yes' ELSE 'No' END`. Below the definition is a table of data with columns: Course, TestDesignerID, and CourseTestDesignersAll. The table lists various courses and their corresponding test designer IDs, with 'Yes' or 'No' indicating the presence of a test designer.

Course	TestDesignerID	CourseTestDesignersAll
1	1	Yes
2	2	Yes
3	3	Yes
4	4	Yes
5	5	Yes
6	6	Yes
7	7	Yes
8	8	Yes
9	9	Yes
10	10	Yes
11	11	Yes
12	12	Yes
13	13	Yes
14	14	Yes
15	15	Yes
16	16	Yes
17	17	Yes
18	18	Yes
19	19	Yes
20	20	Yes
21	21	Yes
22	22	Yes
23	23	Yes
24	24	Yes
25	25	Yes
26	26	Yes
27	27	Yes
28	28	Yes
29	29	Yes
30	30	Yes
31	31	Yes
32	32	Yes
33	33	Yes
34	34	Yes
35	35	Yes
36	36	Yes
37	37	Yes
38	38	Yes
39	39	Yes
40	40	Yes
41	41	Yes
42	42	Yes
43	43	Yes
44	44	Yes
45	45	Yes
46	46	Yes
47	47	Yes
48	48	Yes
49	49	Yes
50	50	Yes
51	51	Yes
52	52	Yes
53	53	Yes
54	54	Yes
55	55	Yes
56	56	Yes
57	57	Yes
58	58	Yes
59	59	Yes
60	60	Yes
61	61	Yes
62	62	Yes
63	63	Yes
64	64	Yes
65	65	Yes
66	66	Yes
67	67	Yes
68	68	Yes
69	69	Yes
70	70	Yes
71	71	Yes
72	72	Yes
73	73	Yes
74	74	Yes
75	75	Yes
76	76	Yes
77	77	Yes
78	78	Yes
79	79	Yes
80	80	Yes
81	81	Yes
82	82	Yes
83	83	Yes
84	84	Yes
85	85	Yes
86	86	Yes
87	87	Yes
88	88	Yes
89	89	Yes
90	90	Yes
91	91	Yes
92	92	Yes
93	93	Yes
94	94	Yes
95	95	Yes
96	96	Yes
97	97	Yes
98	98	Yes
99	99	Yes
100	100	Yes

All testDesigners with at least 1 test.

6. CustomerIn:



The screenshot shows a Tableau workspace with a calculated field named 'CustomerIn' defined as: `IF (COUNT([TicketID]) > 0) THEN 'Yes' ELSE 'No' END`. Below the definition is a table of data with columns: Customer, TicketID, and CustomerIn. The table lists various customers and their corresponding ticket IDs, with 'Yes' or 'No' indicating the presence of a ticket.

Customer	TicketID	CustomerIn
1	1	Yes
2	2	Yes
3	3	Yes
4	4	Yes
5	5	Yes
6	6	Yes
7	7	Yes
8	8	Yes
9	9	Yes
10	10	Yes
11	11	Yes
12	12	Yes
13	13	Yes
14	14	Yes
15	15	Yes
16	16	Yes
17	17	Yes
18	18	Yes
19	19	Yes
20	20	Yes
21	21	Yes
22	22	Yes
23	23	Yes
24	24	Yes
25	25	Yes
26	26	Yes
27	27	Yes
28	28	Yes
29	29	Yes
30	30	Yes
31	31	Yes
32	32	Yes
33	33	Yes
34	34	Yes
35	35	Yes
36	36	Yes
37	37	Yes
38	38	Yes
39	39	Yes
40	40	Yes
41	41	Yes
42	42	Yes
43	43	Yes
44	44	Yes
45	45	Yes
46	46	Yes
47	47	Yes
48	48	Yes
49	49	Yes
50	50	Yes
51	51	Yes
52	52	Yes
53	53	Yes
54	54	Yes
55	55	Yes
56	56	Yes
57	57	Yes
58	58	Yes
59	59	Yes
60	60	Yes
61	61	Yes
62	62	Yes
63	63	Yes
64	64	Yes
65	65	Yes
66	66	Yes
67	67	Yes
68	68	Yes
69	69	Yes
70	70	Yes
71	71	Yes
72	72	Yes
73	73	Yes
74	74	Yes
75	75	Yes
76	76	Yes
77	77	Yes
78	78	Yes
79	79	Yes
80	80	Yes
81	81	Yes
82	82	Yes
83	83	Yes
84	84	Yes
85	85	Yes
86	86	Yes
87	87	Yes
88	88	Yes
89	89	Yes
90	90	Yes
91	91	Yes
92	92	Yes
93	93	Yes
94	94	Yes
95	95	Yes
96	96	Yes
97	97	Yes
98	98	Yes
99	99	Yes
100	100	Yes

All customers with at least 1 ticket.

[illegible][illegible]

The screenshot shows the PyCharm IDE interface. On the left is the Project Structure tool window, displaying the project hierarchy. The main editor window shows a Python script for a 'ReportGenerator' class. The script defines a 'ReportGenerator' class with a 'generate_report' method. The method iterates over a list of 'reports' and generates a report for each. The output of the script is shown in the Run tool window at the bottom, which displays the generated reports for each report ID.

```

class ReportGenerator:
    def __init__(self, reports):
        self.reports = reports

    def generate_report(self):
        for report in self.reports:
            report.generate_report()

# Create a list of reports
reports = [
    Report(1, "Report 1", "Report content 1", "Report date 1"),
    Report(2, "Report 2", "Report content 2", "Report date 2"),
    Report(3, "Report 3", "Report content 3", "Report date 3"),
    Report(4, "Report 4", "Report content 4", "Report date 4"),
    Report(5, "Report 5", "Report content 5", "Report date 5"),
    Report(6, "Report 6", "Report content 6", "Report date 6"),
    Report(7, "Report 7", "Report content 7", "Report date 7"),
    Report(8, "Report 8", "Report content 8", "Report date 8"),
    Report(9, "Report 9", "Report content 9", "Report date 9"),
    Report(10, "Report 10", "Report content 10", "Report date 10"),
    Report(11, "Report 11", "Report content 11", "Report date 11"),
    Report(12, "Report 12", "Report content 12", "Report date 12"),
    Report(13, "Report 13", "Report content 13", "Report date 13"),
    Report(14, "Report 14", "Report content 14", "Report date 14"),
    Report(15, "Report 15", "Report content 15", "Report date 15"),
    Report(16, "Report 16", "Report content 16", "Report date 16"),
    Report(17, "Report 17", "Report content 17", "Report date 17"),
    Report(18, "Report 18", "Report content 18", "Report date 18"),
    Report(19, "Report 19", "Report content 19", "Report date 19"),
    Report(20, "Report 20", "Report content 20", "Report date 20"),
    Report(21, "Report 21", "Report content 21", "Report date 21"),
    Report(22, "Report 22", "Report content 22", "Report date 22"),
    Report(23, "Report 23", "Report content 23", "Report date 23"),
    Report(24, "Report 24", "Report content 24", "Report date 24"),
    Report(25, "Report 25", "Report content 25", "Report date 25"),
    Report(26, "Report 26", "Report content 26", "Report date 26"),
    Report(27, "Report 27", "Report content 27", "Report date 27"),
    Report(28, "Report 28", "Report content 28", "Report date 28"),
    Report(29, "Report 29", "Report content 29", "Report date 29"),
    Report(30, "Report 30", "Report content 30", "Report date 30")
]

# Create a ReportGenerator object
report_generator = ReportGenerator(reports)

# Generate the report
report_generator.generate_report()

```

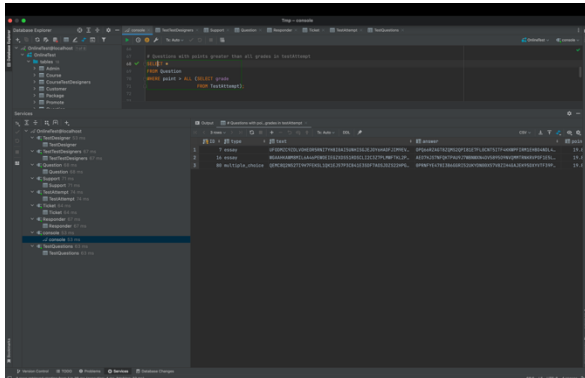
The Run tool window shows the output of the script, displaying the generated reports for each report ID. The output is as follows:

```

Report 1: Report content 1, Report date 1
Report 2: Report content 2, Report date 2
Report 3: Report content 3, Report date 3
Report 4: Report content 4, Report date 4
Report 5: Report content 5, Report date 5
Report 6: Report content 6, Report date 6
Report 7: Report content 7, Report date 7
Report 8: Report content 8, Report date 8
Report 9: Report content 9, Report date 9
Report 10: Report content 10, Report date 10
Report 11: Report content 11, Report date 11
Report 12: Report content 12, Report date 12
Report 13: Report content 13, Report date 13
Report 14: Report content 14, Report date 14
Report 15: Report content 15, Report date 15
Report 16: Report content 16, Report date 16
Report 17: Report content 17, Report date 17
Report 18: Report content 18, Report date 18
Report 19: Report content 19, Report date 19
Report 20: Report content 20, Report date 20
Report 21: Report content 21, Report date 21
Report 22: Report content 22, Report date 22
Report 23: Report content 23, Report date 23
Report 24: Report content 24, Report date 24
Report 25: Report content 25, Report date 25
Report 26: Report content 26, Report date 26
Report 27: Report content 27, Report date 27
Report 28: Report content 28, Report date 28
Report 29: Report content 29, Report date 29
Report 30: Report content 30, Report date 30

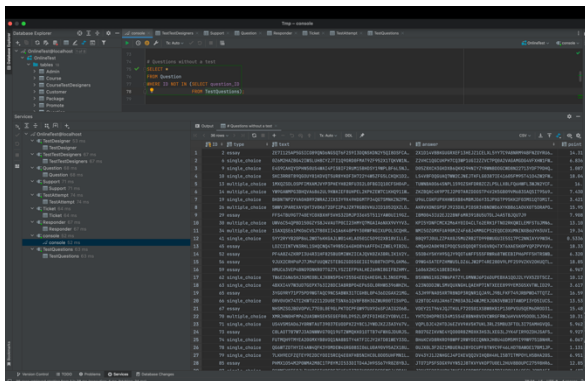
```

10. QuestionAll:



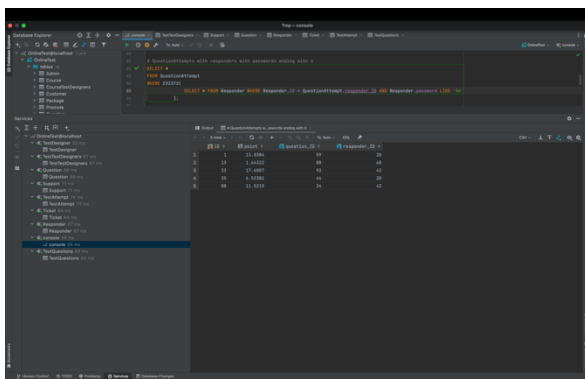
questions with points greater than all grades in testAttempt.

11. QuestionIn:



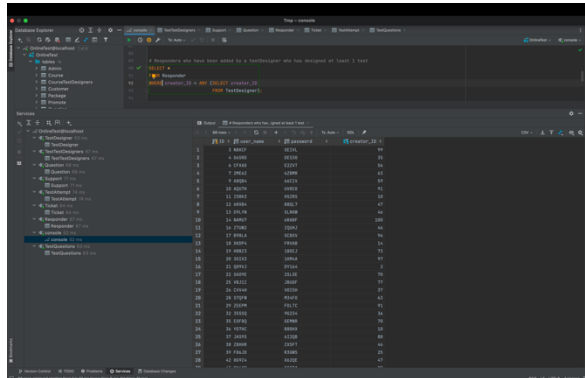
Questions without a test.

12. QuestionAttemptExists:



QuestionAttempts with responders with passwords ending with A.

13. ResponderAny:

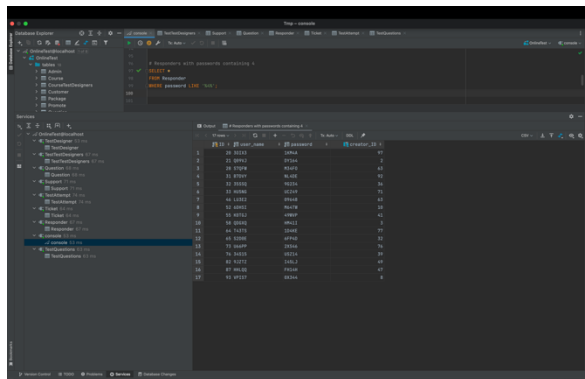


The screenshot shows a data table with columns for various attributes. The data is filtered to show responders who have been added by a testDesigner who has designed at least 1 test.

id	name	password	salary
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
6	6	6	6
7	7	7	7
8	8	8	8
9	9	9	9
10	10	10	10
11	11	11	11
12	12	12	12
13	13	13	13
14	14	14	14
15	15	15	15
16	16	16	16
17	17	17	17
18	18	18	18
19	19	19	19
20	20	20	20
21	21	21	21
22	22	22	22
23	23	23	23
24	24	24	24
25	25	25	25
26	26	26	26
27	27	27	27
28	28	28	28
29	29	29	29
30	30	30	30
31	31	31	31
32	32	32	32
33	33	33	33
34	34	34	34
35	35	35	35
36	36	36	36
37	37	37	37
38	38	38	38
39	39	39	39
40	40	40	40
41	41	41	41
42	42	42	42
43	43	43	43
44	44	44	44
45	45	45	45
46	46	46	46
47	47	47	47
48	48	48	48
49	49	49	49
50	50	50	50
51	51	51	51
52	52	52	52
53	53	53	53
54	54	54	54
55	55	55	55
56	56	56	56
57	57	57	57
58	58	58	58
59	59	59	59
60	60	60	60
61	61	61	61
62	62	62	62
63	63	63	63
64	64	64	64
65	65	65	65
66	66	66	66
67	67	67	67
68	68	68	68
69	69	69	69
70	70	70	70
71	71	71	71
72	72	72	72
73	73	73	73
74	74	74	74
75	75	75	75
76	76	76	76
77	77	77	77
78	78	78	78
79	79	79	79
80	80	80	80
81	81	81	81
82	82	82	82
83	83	83	83
84	84	84	84
85	85	85	85
86	86	86	86
87	87	87	87
88	88	88	88
89	89	89	89
90	90	90	90
91	91	91	91
92	92	92	92
93	93	93	93
94	94	94	94
95	95	95	95
96	96	96	96
97	97	97	97
98	98	98	98
99	99	99	99
100	100	100	100

Responders who have been added by a testDesigner who has designed at least 1 test.

14. ResponderLike:

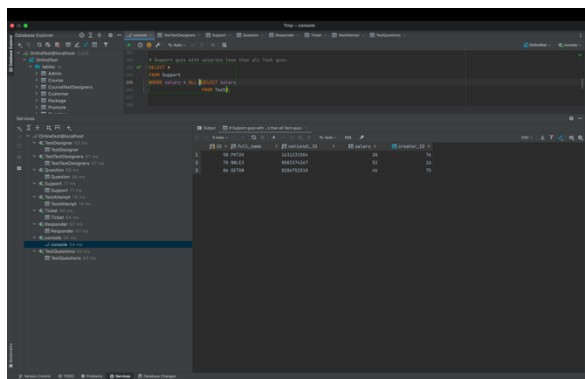


The screenshot shows a data table with columns for various attributes. The data is filtered to show responders with passwords containing 4.

id	name	password	salary
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
6	6	6	6
7	7	7	7
8	8	8	8
9	9	9	9
10	10	10	10
11	11	11	11
12	12	12	12
13	13	13	13
14	14	14	14
15	15	15	15
16	16	16	16
17	17	17	17
18	18	18	18
19	19	19	19
20	20	20	20
21	21	21	21
22	22	22	22
23	23	23	23
24	24	24	24
25	25	25	25
26	26	26	26
27	27	27	27
28	28	28	28
29	29	29	29
30	30	30	30
31	31	31	31
32	32	32	32
33	33	33	33
34	34	34	34
35	35	35	35
36	36	36	36
37	37	37	37
38	38	38	38
39	39	39	39
40	40	40	40
41	41	41	41
42	42	42	42
43	43	43	43
44	44	44	44
45	45	45	45
46	46	46	46
47	47	47	47
48	48	48	48
49	49	49	49
50	50	50	50
51	51	51	51
52	52	52	52
53	53	53	53
54	54	54	54
55	55	55	55
56	56	56	56
57	57	57	57
58	58	58	58
59	59	59	59
60	60	60	60
61	61	61	61
62	62	62	62
63	63	63	63
64	64	64	64
65	65	65	65
66	66	66	66
67	67	67	67
68	68	68	68
69	69	69	69
70	70	70	70
71	71	71	71
72	72	72	72
73	73	73	73
74	74	74	74
75	75	75	75
76	76	76	76
77	77	77	77
78	78	78	78
79	79	79	79
80	80	80	80
81	81	81	81
82	82	82	82
83	83	83	83
84	84	84	84
85	85	85	85
86	86	86	86
87	87	87	87
88	88	88	88
89	89	89	89
90	90	90	90
91	91	91	91
92	92	92	92
93	93	93	93
94	94	94	94
95	95	95	95
96	96	96	96
97	97	97	97
98	98	98	98
99	99	99	99
100	100	100	100

Responders with passwords containing 4.

15. SupportAll:

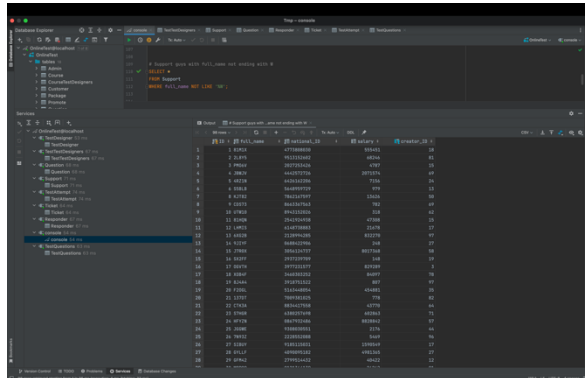


The screenshot shows a data table with columns for various attributes. The data is filtered to show support guys with salaries less than all Tech guys.

id	name	password	salary
1	1	1	1
2	2	2	2
3	3	3	3
4	4	4	4
5	5	5	5
6	6	6	6
7	7	7	7
8	8	8	8
9	9	9	9
10	10	10	10
11	11	11	11
12	12	12	12
13	13	13	13
14	14	14	14
15	15	15	15
16	16	16	16
17	17	17	17
18	18	18	18
19	19	19	19
20	20	20	20
21	21	21	21
22	22	22	22
23	23	23	23
24	24	24	24
25	25	25	25
26	26	26	26
27	27	27	27
28	28	28	28
29	29	29	29
30	30	30	30
31	31	31	31
32	32	32	32
33	33	33	33
34	34	34	34
35	35	35	35
36	36	36	36
37	37	37	37
38	38	38	38
39	39	39	39
40	40	40	40
41	41	41	41
42	42	42	42
43	43	43	43
44	44	44	44
45	45	45	45
46	46	46	46
47	47	47	47
48	48	48	48
49	49	49	49
50	50	50	50
51	51	51	51
52	52	52	52
53	53	53	53
54	54	54	54
55	55	55	55
56	56	56	56
57	57	57	57
58	58	58	58
59	59	59	59
60	60	60	60
61	61	61	61
62	62	62	62
63	63	63	63
64	64	64	64
65	65	65	65
66	66	66	66
67	67	67	67
68	68	68	68
69	69	69	69
70	70	70	70
71	71	71	71
72	72	72	72
73	73	73	73
74	74	74	74
75	75	75	75
76	76	76	76
77	77	77	77
78	78	78	78
79	79	79	79
80	80	80	80
81	81	81	81
82	82	82	82
83	83	83	83
84	84	84	84
85	85	85	85
86	86	86	86
87	87	87	87
88	88	88	88
89	89	89	89
90	90	90	90
91	91	91	91
92	92	92	92
93	93	93	93
94	94	94	94
95	95	95	95
96	96	96	96
97	97	97	97
98	98	98	98
99	99	99	99
100	100	100	100

Support guys with salaries less than all Tech guys.

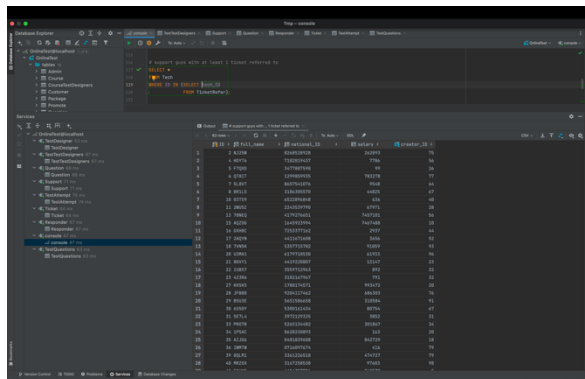
16. SupportLike:



Support guys with full_name not ending with W.

Support guys with full_name not ending with W.

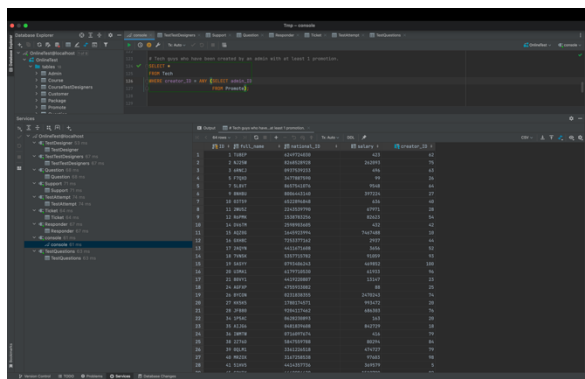
17. TechIn:



support guys with at least 1 ticket referred to.

support guys with at least 1 ticket referred to.

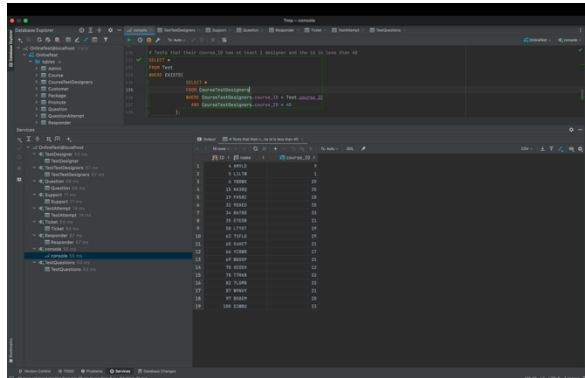
18. TechAny:



Tech guys who have been created by an admin with at least 1 promotion.

Tech guys who have been created by an admin with at least 1 promotion.

19. TestExists:

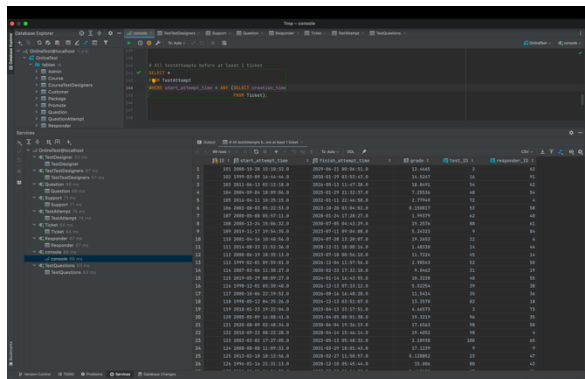


The screenshot shows a SQL query editor with a query that filters for courses where the course_ID has at least one designer and the id is less than 40. The results table shows columns for course_ID, name, and id.

course_ID	name	id
1	1. 00000	1
2	2. 00000	2
3	3. 00000	3
4	4. 00000	4
5	5. 00000	5
6	6. 00000	6
7	7. 00000	7
8	8. 00000	8
9	9. 00000	9
10	10. 00000	10
11	11. 00000	11
12	12. 00000	12
13	13. 00000	13
14	14. 00000	14
15	15. 00000	15
16	16. 00000	16
17	17. 00000	17
18	18. 00000	18
19	19. 00000	19
20	20. 00000	20
21	21. 00000	21
22	22. 00000	22
23	23. 00000	23
24	24. 00000	24
25	25. 00000	25
26	26. 00000	26
27	27. 00000	27
28	28. 00000	28
29	29. 00000	29
30	30. 00000	30
31	31. 00000	31
32	32. 00000	32
33	33. 00000	33
34	34. 00000	34
35	35. 00000	35
36	36. 00000	36
37	37. 00000	37
38	38. 00000	38
39	39. 00000	39
40	40. 00000	40

Tests that their course_ID has at least 1 designer and the id is less than 40.

20. TestAttemptAny:

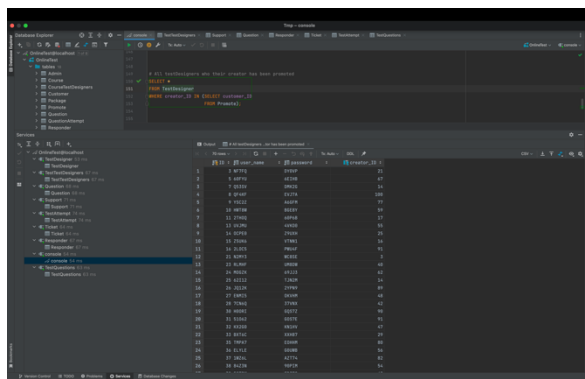


The screenshot shows a SQL query editor with a query that filters for courses where the course_ID has at least one attempt and the id is less than 40. The results table shows columns for course_ID, name, and id.

course_ID	name	id
1	1. 00000	1
2	2. 00000	2
3	3. 00000	3
4	4. 00000	4
5	5. 00000	5
6	6. 00000	6
7	7. 00000	7
8	8. 00000	8
9	9. 00000	9
10	10. 00000	10
11	11. 00000	11
12	12. 00000	12
13	13. 00000	13
14	14. 00000	14
15	15. 00000	15
16	16. 00000	16
17	17. 00000	17
18	18. 00000	18
19	19. 00000	19
20	20. 00000	20
21	21. 00000	21
22	22. 00000	22
23	23. 00000	23
24	24. 00000	24
25	25. 00000	25
26	26. 00000	26
27	27. 00000	27
28	28. 00000	28
29	29. 00000	29
30	30. 00000	30
31	31. 00000	31
32	32. 00000	32
33	33. 00000	33
34	34. 00000	34
35	35. 00000	35
36	36. 00000	36
37	37. 00000	37
38	38. 00000	38
39	39. 00000	39
40	40. 00000	40

All testAttempts before at least 1 ticket.

21. TestDesignerIn:



The screenshot shows a SQL query editor with a query that filters for courses where the course_ID has at least one designer and the id is less than 40. The results table shows columns for course_ID, name, and id.

course_ID	name	id
1	1. 00000	1
2	2. 00000	2
3	3. 00000	3
4	4. 00000	4
5	5. 00000	5
6	6. 00000	6
7	7. 00000	7
8	8. 00000	8
9	9. 00000	9
10	10. 00000	10
11	11. 00000	11
12	12. 00000	12
13	13. 00000	13
14	14. 00000	14
15	15. 00000	15
16	16. 00000	16
17	17. 00000	17
18	18. 00000	18
19	19. 00000	19
20	20. 00000	20
21	21. 00000	21
22	22. 00000	22
23	23. 00000	23
24	24. 00000	24
25	25. 00000	25
26	26. 00000	26
27	27. 00000	27
28	28. 00000	28
29	29. 00000	29
30	30. 00000	30
31	31. 00000	31
32	32. 00000	32
33	33. 00000	33
34	34. 00000	34
35	35. 00000	35
36	36. 00000	36
37	37. 00000	37
38	38. 00000	38
39	39. 00000	39
40	40. 00000	40

All testDesigners whose creator has been promoted.

22. TestDesignerAll:

[illegible]

All testDesigners without a test.

23. TestQuestionsExists:

The screenshot shows the PyCharm IDE interface. On the left, the 'Project' tool window displays the project structure, including a 'src' folder and a 'tests' folder. The 'src' folder contains a 'calendar' module. The 'tests' folder contains a 'test_calendar.py' file. The code editor on the right shows the content of 'test_calendar.py', which is a Python script that tests the 'calendar' module. The script includes a loop that iterates over the years 2019 to 2029, printing the number of days in each month for each year. The script is as follows:

```

import calendar

def test_calendar():
    for year in range(2019, 2029):
        for month in range(1, 12):
            print(f'Year: {year}, Month: {month}, Days: {calendar.monthrange(year, month)[1]}')

if __name__ == '__main__':
    test_calendar()

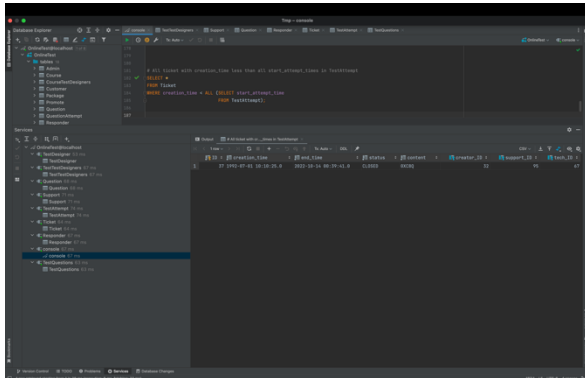
```

All tests without an attempt for at least 1 of its questions.

24. TestTestDesignersAny:

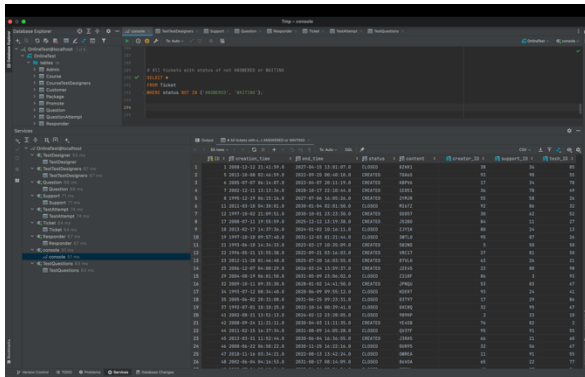
Any test with at least 1 question assigned.

25. TicketAll:



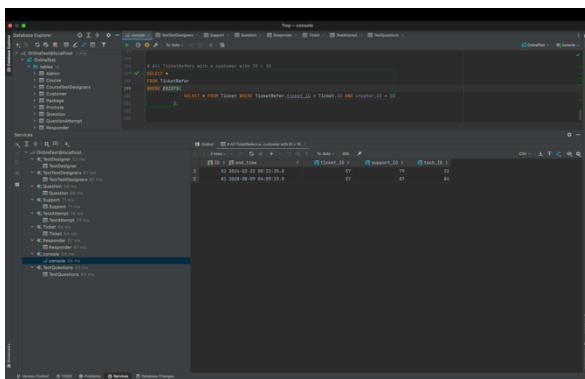
All tickets with creation_time less than all start_attempt_times in TestAttempt.

26. TicketIn:



All tickets with a status of not ANSWERED or WAITING.

27. TicketReferExists:



All TicketRefers with a customer with ID = 10.

0. All Queries:

All queries are in one file, just in case of any error in copying and pasting them in separate files.