

ASSIGNMENT 2

STUDENT MANAGEMENT SYSTEM

Task1:

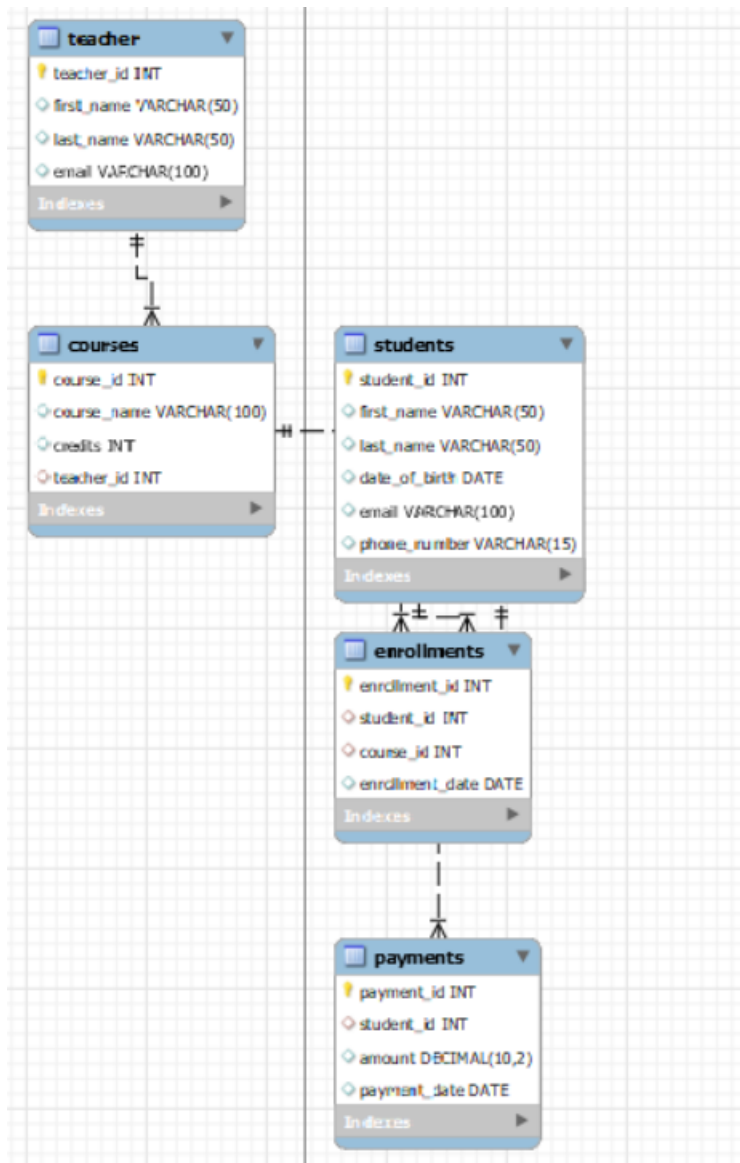
1,2&4.

```
task1.2* x
Limit to 1000 rows

1  /* task1*/
2  ● CREATE DATABASE SISDB;
3  ● USE SISDB;
4
5  ● CREATE TABLE Students (
6      student_id INT PRIMARY KEY,
7      first_name VARCHAR(50),
8      last_name VARCHAR(50),
9      date_of_birth DATE,
10     email VARCHAR(100),
11     phone_number VARCHAR(15)
12 );
13
14 ● CREATE TABLE Teacher (
15     teacher_id INT PRIMARY KEY,
16     first_name VARCHAR(50),
17     last_name VARCHAR(50),
18     email VARCHAR(100)
19 );
20
21 ● CREATE TABLE Courses (
22     course_id INT PRIMARY KEY,
23     course_name VARCHAR(100),
24     credits INT,
25     teacher_id INT,
26     FOREIGN KEY (teacher_id) REFERENCES Teacher(teacher_id)
27 );
28
29 ● CREATE TABLE Enrollments (
30     enrollment_id INT PRIMARY KEY,
31     student_id INT,
32     course_id INT,
33     enrollment_date DATE,
34     FOREIGN KEY (student_id) REFERENCES Students(student_id) ON DELETE CASCADE,
35     FOREIGN KEY (course_id) REFERENCES Courses(course_id)
36 );
37
38 ● CREATE TABLE Payments (
39     payment_id INT PRIMARY KEY,
40     student_id INT,
41     amount DECIMAL(10, 2),
42     payment_date DATE,
43     FOREIGN KEY (student_id) REFERENCES Students(student_id) ON DELETE SET NULL
```

Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
1	16:15:27	CREATE DATABASE SISDB	1 row(s) affected	0.015 sec
2	16:15:30	USE SISDB	0 row(s) affected	0.000 sec
3	16:15:32	CREATE TABLE Students (student_id INT PRIMARY KEY, first_name VARCHAR(50), last_name VARCHAR(50), ...	0 row(s) affected	0.047 sec
4	16:15:35	CREATE TABLE Teacher (teacher_id INT PRIMARY KEY, first_name VARCHAR(50), last_name VARCHAR(50), ...	0 row(s) affected	0.344 sec
5	16:15:40	CREATE TABLE Courses (course_id INT PRIMARY KEY, course_name VARCHAR(100), credits INT, t...	0 row(s) affected	0.047 sec
6	16:15:44	CREATE TABLE Enrollments (enrollment_id INT PRIMARY KEY, student_id INT, course_id INT, enroll...	0 row(s) affected	0.078 sec
7	16:15:47	CREATE TABLE Payments (payment_id INT PRIMARY KEY, student_id INT, amount DECIMAL(10, 2), ...	0 row(s) affected	0.063 sec

3.ERD Diagram.



5.

```
task1.2" x
Limit to 1000 rows

46 ● INSERT INTO Students VALUES
47 (1, 'David', 'Curran', '1990-01-15', 'davidcurran@email.com', '23514269842'),
48 (2, 'Steve', 'Smith', '1992-05-20', 'jane.smith@email.com', '9876543210'),
49 (3, 'Michael', 'Johnson', '1991-08-18', 'michael.johnson@email.com', '5551234567'),
50 (4, 'Sean', 'Williams', '1993-03-25', 'emily.williams@email.com', '1112223333'),
51 (5, 'Daniel', 'Vittori', '1989-11-02', 'daniel.brown@email.com', '9998887777'),
52 (6, 'Olivia', 'Miller', '1995-04-12', 'olivia.miller@email.com', '3334445555'),
53 (7, 'Ethan', 'Davis', '1994-09-18', 'ethan.davis@email.com', '6667778888'),
54 (8, 'Ava', 'Jones', '1992-06-30', 'ava.jones@email.com', '4445556666'),
55 (9, 'Logan', 'Paul', '1990-12-05', 'logan.anderson@email.com', '2223334444'),
56 (10, 'Sophia', 'Moore', '1993-02-28', 'sophia.moore@email.com', '8889990000');
57
58 ● INSERT INTO Teacher VALUES
59 (1, 'Professor', 'Smith', 'prof.smith@email.com'),
60 (2, 'Dr.', 'Johnson', 'dr.johnson@email.com'),
61 (3, 'Ms.', 'Williams', 'ms.williams@email.com'),
62 (4, 'Mr.', 'Davis', 'mr.davis@email.com'),
63 (5, 'Professor', 'Moore', 'prof.moore@email.com'),
64 (6, 'Dr.', 'Anderson', 'dr.anderson@email.com'),
65 (7, 'Mrs.', 'Brown', 'mrs.brown@email.com'),
66 (8, 'Ms.', 'Miller', 'ms.miller@email.com'),
67 (9, 'Mr.', 'Jones', 'mr.jones@email.com'),
68 (10, 'Mrs.', 'Doe', 'mrs.doe@email.com');
69
70 ● INSERT INTO Courses VALUES
71 (101, 'Introduction to Computer Science', 3, 1),
72 (102, 'Mathematics for Engineers', 4, 2),
73 (103, 'History of Art', 3, 3),
74 (104, 'Physics for Beginners', 4, 1),
75 (105, 'Business Ethics', 3, 2),
76 (106, 'Literature and Society', 3, 3),
77 (107, 'Chemistry Fundamentals', 4, 2),
78 (108, 'Psychology 101', 3, 3),
79 (109, 'Data Structures', 4, 1),
80 (110, 'Introduction to Marketing', 3, 2);
```

```
task1.2" x
Limit to 1000 rows

82 ● INSERT INTO Enrollments VALUES
83 (1, 1, 101, '2023-01-01'),
84 (2, 2, 102, '2023-04-02'),
85 (3, 3, 103, '2023-03-03'),
86 (4, 4, 104, '2023-02-04'),
87 (5, 5, 105, '2023-10-05'),
88 (6, 6, 106, '2023-09-06'),
89 (7, 7, 107, '2023-02-07'),
90 (8, 8, 108, '2023-04-08'),
91 (9, 9, 109, '2023-01-09'),
92 (10, 10, 110, '2023-03-10'),
93 (11, 1, 107, '2023-08-05'),
94 (12, 8, 104, '2023-05-12');
95
96 ● INSERT INTO Payments VALUES
97 (1, 1, 500.00, '2023-01-01'),
98 (2, 2, 750.00, '2023-04-02'),
99 (3, 3, 600.00, '2023-03-03'),
100 (4, 4, 800.00, '2023-02-04'),
101 (5, 5, 550.00, '2023-10-05'),
102 (6, 6, 700.00, '2023-09-06'),
103 (7, 7, 850.00, '2023-02-07'),
104 (8, 8, 600.00, '2023-04-08'),
105 (9, 9, 700.00, '2023-01-09'),
106 (10, 10, 500.00, '2023-03-10'),
107 (11, 1, 500.00, '2023-08-01'),
108 (12, 8, 600.00, '2023-05-12');
109
```

Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
1	16:23:19	INSERT INTO Students VALUES (1, 'David', 'Curran', '1990-01-15', 'davidcurran@email.com', '23514269842'), (2, 'Steve', 'Smith', '1992-05-20', 'jane.smith@email.com', '9876543210'), (3, 'Michael', 'Johnson', '1991-08-18', 'michael.johnson@email.com', '5551234567'), (4, 'Sean', 'Williams', '1993-03-25', 'emily.williams@email.com', '1112223333'), (5, 'Daniel', 'Vittori', '1989-11-02', 'daniel.brown@email.com', '9998887777'), (6, 'Olivia', 'Miller', '1995-04-12', 'olivia.miller@email.com', '3334445555'), (7, 'Ethan', 'Davis', '1994-09-18', 'ethan.davis@email.com', '6667778888'), (8, 'Ava', 'Jones', '1992-06-30', 'ava.jones@email.com', '4445556666'), (9, 'Logan', 'Paul', '1990-12-05', 'logan.anderson@email.com', '2223334444'), (10, 'Sophia', 'Moore', '1993-02-28', 'sophia.moore@email.com', '8889990000');	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 sec
2	16:23:31	INSERT INTO Teacher VALUES (1, 'Professor', 'Smith', 'prof.smith@email.com'), (2, 'Dr.', 'Johnson', 'dr.johnson@email.com'), (3, 'Ms.', 'Williams', 'ms.williams@email.com'), (4, 'Mr.', 'Davis', 'mr.davis@email.com'), (5, 'Professor', 'Moore', 'prof.moore@email.com'), (6, 'Dr.', 'Anderson', 'dr.anderson@email.com'), (7, 'Mrs.', 'Brown', 'mrs.brown@email.com'), (8, 'Ms.', 'Miller', 'ms.miller@email.com'), (9, 'Mr.', 'Jones', 'mr.jones@email.com'), (10, 'Mrs.', 'Doe', 'mrs.doe@email.com');	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.047 sec
3	16:23:34	INSERT INTO Courses VALUES (101, 'Introduction to Computer Science', 3, 1), (102, 'Mathematics for Engineers', 4, 2), (103, 'History of Art', 3, 3), (104, 'Physics for Beginners', 4, 1), (105, 'Business Ethics', 3, 2), (106, 'Literature and Society', 3, 3), (107, 'Chemistry Fundamentals', 4, 2), (108, 'Psychology 101', 3, 3), (109, 'Data Structures', 4, 1), (110, 'Introduction to Marketing', 3, 2);	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec
4	16:23:38	INSERT INTO Enrollments VALUES (1, 1, 101, '2023-01-01'), (2, 2, 102, '2023-04-02'), (3, 3, 103, '2023-03-03'), (4, 4, 104, '2023-02-04'), (5, 5, 105, '2023-10-05'), (6, 6, 106, '2023-09-06'), (7, 7, 107, '2023-02-07'), (8, 8, 108, '2023-04-08'), (9, 9, 109, '2023-01-09'), (10, 10, 110, '2023-03-10'), (11, 1, 107, '2023-08-05'), (12, 8, 104, '2023-05-12');	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.016 sec
5	16:23:42	INSERT INTO Payments VALUES (1, 1, 500.00, '2023-01-01'), (2, 2, 750.00, '2023-04-02'), (3, 3, 600.00, '2023-03-03'), (4, 4, 800.00, '2023-02-04'), (5, 5, 550.00, '2023-10-05'), (6, 6, 700.00, '2023-09-06'), (7, 7, 850.00, '2023-02-07'), (8, 8, 600.00, '2023-04-08'), (9, 9, 700.00, '2023-01-09'), (10, 10, 500.00, '2023-03-10'), (11, 1, 500.00, '2023-08-01'), (12, 8, 600.00, '2023-05-12');	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.000 sec

TASK2:

1.

The screenshot shows a SQL IDE window titled 'task1.2'. The main editor contains the following SQL code:

```
112 /*TASK-2*/  
113  
114  
115 • INSERT INTO Students (student_id, first_name, last_name, date_of_birth, email, phone_number)  
116   VALUES (11, 'John', 'Doe', '1995-08-15', 'john.doe@example.com', '1234567890');  
117  
118  
119
```

The right sidebar shows 'SQLAdditions' with 'My Snippets' and 'Context Help' buttons. The bottom 'Output' panel shows the execution result:

#	Time	Action	Message	Duration / Fetch
1	16:25:24	INSERT INTO Students (student_id, first_name, last_name, date_of_birth, email, phone_number) VALUES (11, 'John', 'Doe', '1995-08-15', 'john.doe@example.com', '1234567890');	1 row(s) affected	0.015 sec

2.

The screenshot shows a SQL IDE window titled 'task1.2'. The main editor contains the following SQL code:

```
117  
118 /*2*/  
119 • INSERT INTO Enrollments (enrollment_id, student_id, course_id, enrollment_date)  
120   VALUES (13, 7, 109, '2023-12-10');  
121  
122
```

The right sidebar shows 'SQLAdditions' with 'My Snippets' and 'Context Help' buttons. The bottom 'Output' panel shows the execution result:

#	Time	Action	Message	Duration / Fetch
1	16:27:30	INSERT INTO Enrollments (enrollment_id, student_id, course_id, enrollment_date) VALUES (13, 7, 109, '2023-12-10');	1 row(s) affected	0.000 sec

3.

The screenshot shows a SQL IDE window titled 'task1.2'. The main editor contains the following SQL code:

```
126  
127 /*3*/  
128 • UPDATE Teacher  
129   SET email = 'new.email@example.com'  
130   WHERE teacher_id = 1;  
131
```

The right sidebar shows 'SQLAdditions' with 'My Snippets' and 'Context Help' buttons. The bottom 'Output' panel shows the execution result:

#	Time	Action	Message	Duration / Fetch
1	16:28:22	UPDATE Teacher SET email = 'new.email@example.com' WHERE teacher_id = 1;	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec

4.

The screenshot shows a SQL IDE window titled 'task1.2'. The main editor contains a SQL statement: `DELETE FROM Enrollments WHERE student_id = 1 AND course_id = 101;`. The statement is highlighted in blue. The output pane at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	16:29:24	DELETE FROM Enrollments WHERE student_id = 1 AND course_id = 101	1 row(s) affected	0.016 sec

5.

The screenshot shows a SQL IDE window titled 'task1.2'. The main editor contains a SQL statement: `UPDATE Courses SET teacher_id = 2 WHERE course_id = 105;`. The statement is highlighted in blue. The output pane at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	16:30:39	UPDATE Courses SET teacher_id = 2 WHERE course_id = 105	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.000 sec

6.

The screenshot shows the SQL Developer interface with a script editor and an output window. The script editor contains a SQL statement to delete a student record. The output window shows the successful execution of the statement.

```
150
151
152 /*6*/
153 • DELETE FROM Students
154   WHERE student_id = 4;
155
156
157
```

Output

#	Time	Action	Message	Duration / Fetch
1	16:31:27	DELETE FROM Students WHERE student_id = 4	1 row(s) affected	0.000 sec

7.

The screenshot shows the SQL Developer interface with a script editor and an output window. The script editor contains a SQL statement to update the amount of a payment. The output window shows the successful execution of the statement.

```
157
158
159 /*7*/
160 • UPDATE Payments
161   SET amount = 1500.00
162   WHERE payment_id = 1;
163
164
```

Output

#	Time	Action	Message	Duration / Fetch
1	16:32:12	UPDATE Payments SET amount = 1500.00 WHERE payment_id = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec

TASK3:

1.

The screenshot shows a SQL IDE window titled 'task3'. The SQL editor contains the following query:

```
1  /*Task-3*/
2  /*1*/
3  • SELECT s.student_id, s.first_name, s.last_name, SUM(p.amount) AS total_payments
4  FROM Students s
5  INNER JOIN Payments p ON s.student_id = p.student_id
6  GROUP BY s.student_id, s.first_name, s.last_name;
7
```

The 'Result Grid' shows the following data:

student_id	first_name	last_name	total_payments
1	David	Curran	2000.00
2	Steve	Smith	750.00
3	Michael	Johnson	600.00
5	Daniel	Vittori	550.00
6	Olivia	Miller	700.00
7	Ethan	Davis	850.00
8	Ava	Jones	1200.00
9	Logan	Paul	700.00
10	Sophia	Moore	500.00

The 'Output' pane shows the execution message: 'SELECT s.student_id, s.first_name, s.last_name, SUM(p.amount) AS total_payments FROM Students s INNER JOIN Payments p ON s.student_id = p.student_id GROUP BY s.student_id, s.first_name, s.last_name; 9 row(s) returned'. The duration is 0.000 sec / 0.000 sec.

2.

The screenshot shows a SQL IDE window titled 'task3'. The SQL editor contains the following query:

```
7
8  /*2*/
9  • SELECT c.course_id, c.course_name, COUNT(e.student_id) AS enrolled_students_count
10 FROM Courses c
11 LEFT JOIN Enrollments e ON c.course_id = e.course_id
12 GROUP BY c.course_id, c.course_name;
13
```

The 'Result Grid' shows the following data:

course_id	course_name	enrolled_students_count
101	Introduction to Computer Science	0
102	Mathematics for Engineers	1
103	History of Art	1
104	Physics for Beginners	1
105	Business Ethics	1
106	Literature and Society	1
107	Chemistry Fundamentals	2
108	Psychology 101	1
109	Data Structures	2

The 'Output' pane shows the execution message: 'SELECT c.course_id, c.course_name, COUNT(e.student_id) AS enrolled_students_count FROM Courses c LEFT JOIN Enrollments e ON c.course_id = e.course_id GROUP BY c.course_id, c.course_name; 10 row(s) returned'. The duration is 0.000 sec / 0.000 sec.

3.

The screenshot shows a SQL IDE window titled 'task3'. The query editor contains the following SQL code:

```
13
14 /*3*/
15 SELECT s.first_name, s.last_name
16 FROM Students s
17 LEFT JOIN Enrollments e ON s.student_id = e.student_id
18 WHERE e.enrollment_id IS NULL;
19
```

The 'Result Grid' shows one row of data:

first_name	last_name
John	Doe

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:36:17	SELECT s.first_name, s.last_name FROM Students s LEFT JOIN Enrollments e ON s.student_id = e.student_id ...	1 row(s) returned	0.000 sec / 0.000 sec

4.

The screenshot shows a SQL IDE window titled 'task3'. The query editor contains the following SQL code:

```
19
20 /*4*/
21 SELECT s.first_name, s.last_name, c.course_name
22 FROM Students s
23 JOIN Enrollments e ON s.student_id = e.student_id
24 JOIN Courses c ON e.course_id = c.course_id;
25
```

The 'Result Grid' shows 11 rows of data:

first_name	last_name	course_name
David	Curran	Chemistry Fundamentals
Steve	Smith	Mathematics for Engineers
Michael	Johnson	History of Art
Daniel	Vittori	Business Ethics
Olivia	Miller	Literature and Society
Ethan	Davis	Chemistry Fundamentals
Ethan	Davis	Data Structures
Ava	Jones	Psychology 101
Ava	Jones	Physics for Beginners

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:36:54	SELECT s.first_name, s.last_name, c.course_name FROM Students s JOIN Enrollments e ON s.student_id = e.st...	11 row(s) returned	0.000 sec / 0.000 sec

5.

The screenshot shows a database IDE window titled 'task3*' with a toolbar at the top. The SQL editor contains the following query:

```
26  
27 /*5*/  
28 • SELECT t.first_name, t.last_name, c.course_name  
29 FROM Teacher t  
30 JOIN Courses c ON t.teacher_id = c.teacher_id;  
31  
32
```

Below the editor is the 'Result Grid' showing 6 rows of data:

first_name	last_name	course_name
Professor	Smith	Introduction to Computer Science
Professor	Smith	Physics for Beginners
Professor	Smith	Data Structures
Dr.	Johnson	Mathematics for Engineers
Dr.	Johnson	Business Ethics
Dr.	Johnson	Chemistry Fundamentals
Dr.	Johnson	Introduction to Marketing
Ms.	Williams	History of Art
Ms.	Williams	Literature and Society

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:37:53	SELECT t.first_name, t.last_name, c.course_name FROM Teacher t JOIN Courses c ON t.teacher_id = c.teacher_id	10 row(s) returned	0.000 sec / 0.000 sec

6.

The screenshot shows a database IDE window titled 'task3*' with a toolbar at the top. The SQL editor contains the following query:

```
33  
34 /*6*/  
35 • SELECT s.first_name, s.last_name, e.enrollment_date  
36 FROM Students s  
37 JOIN Enrollments e ON s.student_id = e.student_id;  
38  
39
```

Below the editor is the 'Result Grid' showing 8 rows of data:

first_name	last_name	enrollment_date
David	Curran	2023-08-05
Steve	Smith	2023-04-02
Michael	Johnson	2023-03-03
Daniel	Vittori	2023-10-05
Olivia	Miller	2023-09-06
Ethan	Davis	2023-02-07
Ethan	Davis	2023-12-10
Ava	Jones	2023-04-08
Ava	Jones	2023-05-12

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:38:28	SELECT s.first_name, s.last_name, e.enrollment_date FROM Students s JOIN Enrollments e ON s.student_id = e.student_id	11 row(s) returned	0.000 sec / 0.000 sec

7.

The screenshot shows the SQL Developer interface with a query window titled 'task3'. The query is as follows:

```
39
40 /*7*/
41 • SELECT s.first_name, s.last_name
42 FROM Students s
43 LEFT JOIN Payments p ON s.student_id = p.student_id
44 WHERE p.payment_id IS NULL;
45
```

The 'Result Grid' shows the following data:

first_name	last_name
John	Doe

The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:38:28	SELECT s.first_name, s.last_name, e.enrollment_date FROM Students s JOIN Enrollments e ON s.student_id = e...	11 row(s) returned	0.000 sec / 0.000 sec
2	16:39:11	SELECT s.first_name, s.last_name FROM Students s LEFT JOIN Payments p ON s.student_id = p.student_id W...	1 row(s) returned	0.000 sec / 0.000 sec

8.

The screenshot shows the SQL Developer interface with a query window titled 'task3'. The query is as follows:

```
45
46 /*8*/
47 • SELECT c.course_id, c.course_name
48 FROM Courses c
49 LEFT JOIN Enrollments e ON c.course_id = e.course_id
50 WHERE e.enrollment_id IS NULL;
51
```

The 'Result Grid' shows the following data:

course_id	course_name
101	Introduction to Computer Science

The 'Output' pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	16:39:52	SELECT c.course_id, c.course_name FROM Courses c LEFT JOIN Enrollments e ON c.course_id = e.course_id ...	1 row(s) returned	0.000 sec / 0.000 sec

9.

task3

```

51
52 /*9*/
53 • SELECT e1.student_id, count(e1.student_id) AS no_of_enrollments
54 FROM Enrollments e1
55 JOIN Enrollments e2 ON e1.student_id = e2.student_id AND e1.enrollment_id <> e2.enrollment_id
56 GROUP BY e1.student_id HAVING COUNT(DISTINCT e2.course_id) > 1;
57

```

SQLAdditions: My Snippets

Result Grid

student_id	no_of_enrollments
7	2
8	2

Result 10

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:40:38	SELECT e1 student_id, count(e1 student_id) AS no_of_enrollments FROM Enrollments e1 JOIN Enrollments e2 ...	2 row(s) returned	0.000 sec / 0.000 sec

10.

task3

```

58
59 /*10*/
60 • SELECT t.teacher_id, t.first_name, t.last_name
61 FROM Teacher t
62 LEFT JOIN Courses c ON t.teacher_id = c.teacher_id
63 WHERE c.course_id IS NULL;
64

```

SQLAdditions: My Snippets

Result Grid

teacher_id	first_name	last_name
4	Mr.	Davis
5	Professor	Moore
6	Dr.	Anderson
7	Mrs.	Brown
8	Ms.	Miller
9	Mr.	Jones
10	Mrs.	Doe

Result 11

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	16:41:10	SELECT t.teacher_id, t.first_name, t.last_name FROM Teacher t LEFT JOIN Courses c ON t.teacher_id = c.teac...	7 row(s) returned	0.000 sec / 0.000 sec

Task 4:

1.

The screenshot shows the SQL Developer interface with a query window titled 'task4'. The query is as follows:

```
1  /*Task-4*/
2  • SELECT course_id, AVG(student_count) AS avg_students_enrolled
3  FROM (
4      SELECT course_id, COUNT(student_id) AS student_count
5      FROM Enrollments
6      GROUP BY course_id
7  ) AS course_enrollment_counts
8  GROUP BY course_id;
```

The 'Result Grid' shows the following data:

course_id	avg_students_enrolled
102	1.0000
103	1.0000
104	1.0000
105	1.0000
106	1.0000
107	2.0000
108	1.0000
109	2.0000

The 'Output' pane shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	13:05:01	SELECT course_id, AVG(student_count) AS avg_students_enrolled FROM (SELECT course_id, COUNT(stud...	9 row(s) returned	0.094 sec / 0.000 sec

2.

The screenshot shows the SQL Developer interface with a query window titled 'task4'. The query is as follows:

```
10  /*2*/
11  • SELECT student_id, first_name, last_name
12  FROM Students
13  WHERE student_id = (
14      SELECT student_id
15      FROM Payments
16      ORDER BY amount DESC LIMIT 1
17  );
```

The 'Result Grid' shows the following data:

student_id	first_name	last_name
1	David	Curran

The 'Output' pane shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	13:05:07	SELECT student_id, first_name, last_name FROM Students WHERE student_id = (SELECT student_id FR...	1 row(s) returned	0.469 sec / 0.000 sec

3.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query is as follows:

```
19  /*3*/
20  • SELECT course_id, course_name, enrollment_count
21  FROM (
22      SELECT C.course_id, C.course_name, COUNT(E.student_id) AS enrollment_count
23      FROM Courses C
24      JOIN Enrollments E ON C.course_id = E.course_id
25      GROUP BY C.course_id, C.course_name
26  ) AS course_enrollment_counts
27  ORDER BY enrollment_count DESC;
28
```

The result grid displays the following data:

course_id	course_name	enrollment_count
107	Chemistry Fundamentals	2
109	Data Structures	2
102	Mathematics for Engineers	1
103	History of Art	1
105	Business Ethics	1
106	Literature and Society	1
101	Psychology 101	1

The output pane shows the execution of the query, indicating that 9 rows were returned.

4.

The screenshot shows a SQL IDE window with a query editor and a result grid. The query is as follows:

```
29  /*4*/
30  • SELECT teacher_id, SUM(amount) AS total_payments
31  FROM (
32      SELECT T.teacher_id, P.amount
33      FROM Teacher T
34      JOIN Courses C ON T.teacher_id = C.teacher_id
35      JOIN Enrollments E ON C.course_id = E.course_id
36      JOIN Payments P ON E.student_id = P.student_id
37  ) AS teacher_payments
38  GROUP BY teacher_id;
```

The result grid displays the following data:

teacher_id	total_payments
2	4650.00
3	2500.00
1	2750.00

The output pane shows the execution of the query, indicating that 3 rows were returned.

5.

```
40
41  /*5*/
42 • SELECT student_id, first_name, last_name
43 FROM Students
44 WHERE (SELECT COUNT(DISTINCT course_id) FROM Courses) = (
45     SELECT COUNT(DISTINCT course_id)
46     FROM Enrollments
47     WHERE Students.student_id = Enrollments.student_id
48 );
49
```

student_id	first_name	last_name
1
2
3
4
5
6
7

Students 7 x

Output

#	Time	Action	Message	Duration / Fetch
1	13:08:35	SELECT student_id, first_name, last_name FROM Students WHERE (SELECT COUNT(DISTINCT course_id) F...	0 row(s) returned	0.109 sec / 0.000 sec

6.

```
50
51
52  /*6*/
53 • SELECT teacher_id, first_name, last_name
54 FROM Teacher
55 WHERE teacher_id NOT IN (
56     SELECT DISTINCT teacher_id FROM Courses
57 );
58
59
```

teacher_id	first_name	last_name
4	Mr.	Davis
5	Professor	Moore
6	Dr.	Anderson
7	Mrs.	Brown
8	Ms.	Miller
9	Mr.	Jones
10	Mr.	Pine

Teacher 8 x

Output

#	Time	Action	Message	Duration / Fetch
1	13:09:10	SELECT teacher_id, first_name, last_name FROM Teacher WHERE teacher_id NOT IN (SELECT DISTINCT...	7 row(s) returned	0.015 sec / 0.000 sec

7.

The screenshot shows a database IDE window titled 'task4'. The SQL editor contains the following query:

```
58
59
60 /*7*/
61 • SELECT AVG(age) AS average_age
62 FROM (
63     SELECT student_id, TIMESTAMPDIFF(YEAR, date_of_birth, CURRENT_DATE()) AS age
64     FROM Students
65 ) AS student_age;
66
67
```

Below the editor, the 'Result Grid' shows a single row with the value 30.9000 for the column 'average_age'.

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	13:14:00	SELECT AVG(age) AS average_age FROM (SELECT student_id, TIMESTAMPDIFF(YEAR, date_of_birth, C...	1 row(s) returned	0.000 sec / 0.000 sec

8.

The screenshot shows a database IDE window titled 'task4'. The SQL editor contains the following query:

```
67
68
69 /*8*/
70 • SELECT course_id, course_name
71 FROM Courses
72 WHERE course_id NOT IN (
73     SELECT DISTINCT course_id FROM Enrollments
74 );
75
76
```

Below the editor, the 'Result Grid' shows a single row with the values 101 and Introduction to Computer Science for the columns 'course_id' and 'course_name' respectively.

The 'Output' pane at the bottom shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	13:14:37	SELECT course_id, course_name FROM Courses WHERE course_id NOT IN (SELECT DISTINCT course_id...	1 row(s) returned	0.000 sec / 0.000 sec

9.

task4*

Limit to 1000 rows

```

76
77 /*9*/
78 • SELECT E.student_id, E.course_id, IFNULL(SUM(P.amount), 0) AS total_payments
79 FROM Enrollments E
80 LEFT JOIN Payments P ON E.student_id = P.student_id
81 WHERE E.student_id IN (SELECT DISTINCT student_id FROM Enrollments)
82 GROUP BY E.student_id, E.course_id;
83
84
85

```

Result Grid

student_id	course_id	total_payments
1	107	2000.00
2	102	750.00
3	103	600.00
5	105	550.00
6	106	700.00
7	107	850.00
9	108	850.00

Result 12 x

Read Only Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:16:09	SELECT E.student_id, E.course_id, IFNULL(SUM(P.amount), 0) AS total_payments FROM Enrollments E LEFT J...	11 row(s) returned	0.000 sec / 0.000 sec

10.

task4*

Limit to 1000 rows

```

84 /*10*/
85 • SELECT student_id, first_name, last_name
86 FROM Students
87 WHERE student_id IN (
88     SELECT student_id
89     FROM Payments
90     GROUP BY student_id
91     HAVING COUNT(payment_id) > 1
92 );
93

```

Result Grid

student_id	first_name	last_name
1	David	Curran
8	Ava	Jones
9	NULL	NULL

Students 13 x

Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	13:16:56	SELECT student_id, first_name, last_name FROM Students WHERE student_id IN (SELECT student_id F...	2 row(s) returned	0.000 sec / 0.000 sec

11.

The screenshot shows the SQL Developer interface with a query window titled 'task4'. The query is as follows:

```
93  
94  
95 /*11*/  
96 • SELECT S.student_id, S.first_name, S.last_name, SUM(P.amount) AS total_payments  
97 FROM Students S  
98 LEFT JOIN Payments P ON S.student_id = P.student_id  
99 GROUP BY S.student_id, S.first_name, S.last_name;  
100  
101  
102
```

The 'Result Grid' shows the following data:

student_id	first_name	last_name	total_payments
1	David	Curran	2000.00
2	Steve	Smith	750.00
3	Michael	Johnson	600.00
5	Daniel	Vittori	550.00
6	Olivia	Miller	700.00
7	Ethan	Davis	850.00
8	Ava	Donner	1200.00

The 'Output' window shows the execution message: '1 13:17:30 SELECT S.student_id, S.first_name, S.last_name, SUM(P.amount) AS total_payments FROM Students S LEFT J... 10 row(s) returned'.

12.

The screenshot shows the SQL Developer interface with a query window titled 'task4'. The query is as follows:

```
100  
101  
102  
103  
104 /*12*/  
105 • SELECT C.course_id, C.course_name, COUNT(E.student_id) AS enrolled_students_count  
106 FROM Courses C  
107 LEFT JOIN Enrollments E ON C.course_id = E.course_id  
108 GROUP BY C.course_id, C.course_name;  
109
```

The 'Result Grid' shows the following data:

course_id	course_name	enrolled_students_count
101	Introduction to Computer Science	0
102	Mathematics for Engineers	1
103	History of Art	1
104	Physics for Beginners	1
105	Business Ethics	1
106	Literature and Society	1
107	Chemistry Fundamentals	2

The 'Output' window shows the execution message: '1 13:18:12 SELECT C.course_id, C.course_name, COUNT(E.student_id) AS enrolled_students_count FROM Courses C LE... 10 row(s) returned'.

13.

The screenshot shows a SQL IDE interface with a query editor, a result grid, and an output pane.

Query Editor: The query is as follows:

```
110
111
112
113 /*13*/
114 • SELECT AVG(subquery.amount) AS average_payment_amount
115 FROM (
116     SELECT P.amount
117     FROM Payments P
118     JOIN Students S ON P.student_id = S.student_id
119 ) AS subquery;
```

Result Grid: The result grid shows a single column named `average_payment_amount` with a single row containing the value `713.636364`.

Output Pane: The output pane shows the execution of the query. The message is: `1 13:20:06 SELECT AVG(subquery.amount) AS average_payment_amount FROM (SELECT P.amount FROM Paymen... 1 row(s) returned`. The duration is `0.000 sec / 0.000 sec`.