

# Assignment 3

## Banking system

### Task1:

1,2,5&6.

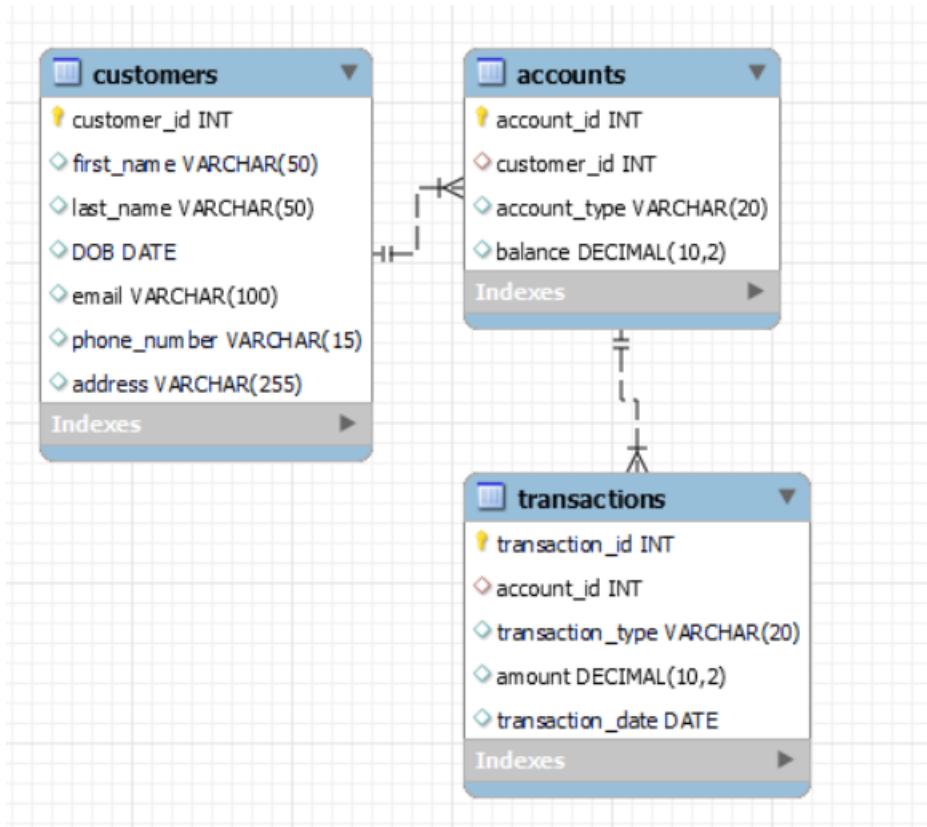
The screenshot shows the MySQL Workbench interface with a query editor titled "task 1\*". The code in the editor creates a database named "HMBank" and three tables: "Customers", "Accounts", and "Transactions". The "Customers" table has columns for customer\_id (primary key), first\_name, last\_name, DOB, email, phone\_number, and address. The "Accounts" table has columns for account\_id (primary key), customer\_id, account\_type, and balance, with a foreign key constraint linking customer\_id to the primary key of Customers. The "Transactions" table has columns for transaction\_id (primary key), account\_id, transaction\_type, amount, and transaction\_date, with a foreign key constraint linking account\_id to the primary key of Accounts.

```
1  /*TASK-1*/
2  ● CREATE DATABASE HMBank;
3  ● USE HMBank;
4
5  ● CREATE TABLE Customers (
6      customer_id INT PRIMARY KEY,
7      first_name VARCHAR(50),
8      last_name VARCHAR(50),
9      DOB DATE,
10     email VARCHAR(100),
11     phone_number VARCHAR(15),
12     address VARCHAR(255)
13 );
14
15 ● CREATE TABLE Accounts (
16     account_id INT PRIMARY KEY,
17     customer_id INT,
18     account_type VARCHAR(20),
19     balance DECIMAL(10, 2),
20     FOREIGN KEY (customer_id) REFERENCES Customers(customer_id) ON DELETE SET NULL
21 );
22
23 ● CREATE TABLE Transactions (
24     transaction_id INT PRIMARY KEY,
25     account_id INT,
26     transaction_type VARCHAR(20),
27     amount DECIMAL(10, 2),
28     transaction_date DATE,
29     FOREIGN KEY (account_id) REFERENCES Accounts(account_id) ON DELETE SET NULL
30 );
```

The "Output" tab shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	17:00:02	CREATE DATABASE HMBank	1 row(s) affected	0.016 sec
2	17:00:09	USE HMBank	0 row(s) affected	0.000 sec
3	17:00:11	CREATE TABLE Customers ( customer_id INT PRIMARY KEY, first_name VARCHAR(50), last_name VAR... )	0 row(s) affected	0.047 sec
4	17:00:15	CREATE TABLE Accounts ( account_id INT PRIMARY KEY, customer_id INT, account_type VARCHAR(... ) )	0 row(s) affected	0.094 sec
5	17:00:19	CREATE TABLE Transactions ( transaction_id INT PRIMARY KEY, account_id INT, transaction_type VA... )	0 row(s) affected	0.078 sec

### 3.ER Diagram



## Task2:

### 1.

task 1\* x

31    /task 2 1st part\*/  
32 ●    INSERT INTO Customers VALUES  
33    (1, 'Rahul', 'Sharma', '1990-05-15', 'rahul.sharma@email.com', '9876543210', 'Delhi'),  
34    (2, 'Priya', 'Patel', '1985-08-22', 'priya.patel@email.com', '8765432109', 'Mumbai'),  
35    (3, 'Amit', 'Singh', '1988-04-12', 'amit.singh@email.com', '7654321098', 'Kolkata'),  
36    (4, 'Deepa', 'Shah', '1995-09-28', 'deepa.shah@email.com', '6543210987', 'Chennai'),  
37    (5, 'Rajesh', 'Kumar', '1977-12-05', 'rajesh.kumar@email.com', '5432109876', 'Hyderabad'),  
38    (6, 'Anita', 'Gupta', '1982-03-20', 'anita.gupta@email.com', '4321098765', 'Bangalore'),  
39    (7, 'Suresh', 'Verma', '1992-06-18', 'suresh.verma@email.com', '3210987654', 'Mumbai'),  
40    (8, 'Preeti', 'Agarwal', '1980-11-15', 'preeti.agarwal@email.com', '2109876543', 'Delhi'),  
41    (9, 'Vikram', 'Rajput', '1993-02-25', 'vikram.rajput@email.com', '1098765432', 'Chandigarh'),  
42    (10, 'Neha', 'Sharma', '1986-07-30', 'neha.sharma@email.com', '9876543210', 'Jaipur');  
43  
44 ●    INSERT INTO Accounts VALUES  
45    (101, 1, 'savings', 5000.00),  
46    (102, 10, 'zero\_balance', 10000.00),  
47    (103, 2, 'savings', 8000.00),  
48    (104, 3, 'current', 12000.00),  
49    (105, 4, 'savings', 6000.00),  
50    (106, 5, 'zero\_balance', 15000.00),  
51    (107, 6, 'savings', 7000.00),  
52    (108, 7, 'current', 18000.00),  
53    (109, 8, 'zero\_balance', 9000.00),  
54    (110, 9, 'current', 20000.00);  
55  
56 ●    INSERT INTO Transactions VALUES  
57    (1001, 101, 'deposit', 2000.00, '2023-01-10'),  
58    (1002, 101, 'withdrawal', 1000.00, '2023-02-15'),  
59    (1003, 107, 'deposit', 3000.00, '2023-03-20'),  
60    (1004, 102, 'withdrawal', 1500.00, '2023-04-25'),  
61    (1005, 104, 'deposit', 2500.00, '2023-05-10'),  
62    (1006, 103, 'withdrawal', 1200.00, '2023-06-15'),  
63    (1007, 104, 'deposit', 4000.00, '2023-07-05'),  
64    (1008, 105, 'withdrawal', 2000.00, '2023-08-18'),  
65    (1009, 105, 'deposit', 3500.00, '2023-09-22'),  
66    (1010, 105, 'withdrawal', 1800.00, '2023-10-30');

Output				
#	Time	Action	Message	Duration / Fetch
1	17:05:51	INSERT INTO Customers VALUES (1, 'Rahul', 'Sharma', '1990-05-15', 'rahul.sharma@email.com', '9876543210', 'Delhi')	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 sec
2	17:05:53	INSERT INTO Accounts VALUES (101, 1, 'savings', 5000.00), (102, 10, 'zero_balance', 10000.00), (103, 2, 'savings', 8000.00), (104, 3, 'current', 12000.00), (105, 4, 'savings', 6000.00), (106, 5, 'zero_balance', 15000.00), (107, 6, 'savings', 7000.00), (108, 7, 'current', 18000.00), (109, 8, 'zero_balance', 9000.00), (110, 9, 'current', 20000.00)	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 sec
3	17:05:55	INSERT INTO Transactions VALUES (1001, 101, 'deposit', 2000.00, '2023-01-10'), (1002, 101, 'withdrawal', 1000.00, '2023-02-15'), (1003, 107, 'deposit', 3000.00, '2023-03-20'), (1004, 102, 'withdrawal', 1500.00, '2023-04-25'), (1005, 104, 'deposit', 2500.00, '2023-05-10'), (1006, 103, 'withdrawal', 1200.00, '2023-06-15'), (1007, 104, 'deposit', 4000.00, '2023-07-05'), (1008, 105, 'withdrawal', 2000.00, '2023-08-18'), (1009, 105, 'deposit', 3500.00, '2023-09-22'), (1010, 105, 'withdrawal', 1800.00, '2023-10-30')	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.016 sec

## TASK 2(2):

1.

The screenshot shows the SQL Server Management Studio interface with a query window titled "task 2". The query is:

```
1 /*task 2 2nd part*/
2 /*
3 • select first_name, last_name, email, account_type from customers c
4 left join accounts a on a.customer_id=c.customer_id;
5
```

The results grid displays the following data:

first_name	last_name	email	account_type
Rahul	Sharma	rahul.sharma@email.com	savings
Priya	Patel	priya.patel@email.com	savings
Amit	Singh	amit.singh@email.com	current
Deepa	Shah	deepa.shah@email.com	savings
Rajesh	Kumar	rajesh.kumar@email.com	zero_balance
Anita	Gupta	anita.gupta@email.com	savings
Suresh	Verma	suresh.verma@email.com	current
Preeti	Agarwal	preeti.agarwal@email.com	zero_balance
Vikram	Raput	vikram.raput@email.com	current
Neha	Sharma	neha.sharma@email.com	zero_balance

The output pane shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	17:07:50	select first_name,last_name,email,account_type from customers c left join accounts a on a.customer_id=c.custo...	10 row(s) returned	0.000 sec / 0.000 sec

2.

The screenshot shows the SQL Server Management Studio interface with a query window titled "task 2". The query is:

```
7 /*2*/
8 • select t.transaction_id,t.transaction_type,t.amount,c.customer_id from transactions t
9 left join accounts a on a.account_id=t.account_id
10 left join customers c on c.customer_id=a.customer_id;
11
```

The results grid displays the following data:

transaction_id	transaction_type	amount	customer_id
1001	deposit	2000.00	1
1002	withdrawal	1000.00	1
1003	deposit	3000.00	6
1004	withdrawal	1500.00	10
1005	deposit	2500.00	3
1006	withdrawal	1200.00	2
1007	deposit	4000.00	3
1008	withdrawal	2000.00	8
1009	deposit	3500.00	4
1010	withdrawal	1800.00	4

The output pane shows the execution details:

#	Time	Action	Message	Duration / Fetch
1	17:07:50	select first_name,last_name,email,account_type from customers c left join accounts a on a.customer_id=c.custo...	10 row(s) returned	0.000 sec / 0.000 sec
2	17:09:05	select t.transaction_id,t.transaction_type,t.amount,c.customer_id from transactions t left join accounts a on a.ac...	10 row(s) returned	0.000 sec / 0.000 sec

### 3.

Screenshot of SQL Server Management Studio (task 2) showing an UPDATE query:

```
12  /*3*/
13 • UPDATE Accounts
14   SET balance = balance + 200
15   WHERE account_id = 101;
16
```

The Output window shows the execution results:

#	Time	Action
1	17:10:22	UPDATE Accounts SET balance = balance + 200 WHERE account_id = 101

Message: 1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0 Duration / Fetch: 0.016 sec

### 4.

Screenshot of SQL Server Management Studio (task 2) showing a SELECT query:

```
18
19  /*4*/
20 • SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM Customers;
21
```

The Results Grid displays the following data:

full_name
Rahul Sharma
Priya Patel
Amit Singh
Deepa Shah
Rajesh Kumar
Anita Gupta
Suresh Verma
Preeti Agarwal

The Output window shows the execution results:

#	Time	Action
1	17:11:05	SELECT CONCAT(first_name, '' ,last_name) AS full_name FROM Customers LIMIT 0, 1000

Message: 10 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec

### 5.

Screenshot of SQL Server Management Studio (task 2) showing a DELETE query:

```
23  /*5*/
24 • DELETE FROM Accounts
25   WHERE balance = 0 AND account_type = 'savings';
```

The Output window shows the execution results:

#	Time	Action
1	17:12:30	DELETE FROM Accounts WHERE balance = 0 AND account_type = 'savings'

Message: 0 row(s) affected Duration / Fetch: 0.000 sec

6.

The screenshot shows a SQL query window titled "task 2" with the following code:

```
26
27      /*6*/
28 •  SELECT * FROM Customers WHERE address = 'Mumbai';
29
```

The results grid displays two rows of customer data:

customer_id	first_name	last_name	DOB	email	phone_number	address
2	Priya	Patel	1985-08-22	priya.patel@email.com	8765432109	Mumbai
7	Suresh	Verma	1992-06-18	suresh.verma@email.com	3210987654	Mumbai
•	NULL	NULL	NULL	NULL	NULL	NULL

The output pane shows the query and its execution details:

#	Time	Action
1	17:13:11	SELECT * FROM Customers WHERE address = 'Mumbai' LIMIT 0, 1000

Message: 2 row(s) returned  
Duration / Fetch: 0.000 sec / 0.000 sec

7.

The screenshot shows a SQL query window titled "task 2" with the following code:

```
30
31      /*7*/
32 •  SELECT balance
33     FROM Accounts
34    WHERE account_id = 101;
35
```

The results grid displays one row of account data:

balance
5200.00

The output pane shows the query and its execution details:

#	Time	Action
1	17:14:35	SELECT balance FROM Accounts WHERE account_id = 101 LIMIT 0, 1000

Message: 1 row(s) returned  
Duration / Fetch: 0.000 sec / 0.000 sec

8.

task 2\* ×

```
36
37  /*8*/
38 • SELECT account_id, balance
39   FROM Accounts
40  WHERE account_type = 'current' AND balance > 1000.00;
41
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

account_id	balance
104	12000.00
108	18000.00
210	20000.00
NULL	NULL

SQLAdditions ··· My Snippets ···

Accounts 7 ×

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:15:59	SELECT account_id, balance FROM Accounts WHERE account_type = 'current' AND balance > 1000.00 LIMIT 0, 1000	3 row(s) returned	0.000 sec / 0.000 sec

Context Help ··· Snippets ···

9.

task 2\* ×

```
42
43  /*9*/
44 • SELECT *
45   FROM Transactions
46  WHERE account_id = 101;
47
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

transaction_id	account_id	transaction_type	amount	transaction_date
1001	101	deposit	2000.00	2023-01-10
1002	101	withdrawal	1000.00	2023-02-15
NULL	NULL	NULL	NULL	NULL

SQLAdditions ··· My Snippets ···

Transactions 8 ×

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:16:32	SELECT * FROM Transactions WHERE account_id = 101 LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

Context Help ··· Snippets ···

## 10.

The screenshot shows a SQL database interface with a query editor and a results grid. The query is:

```
48
49  /*10*/
50 • SELECT account_id, balance * 0.05 AS interest_accrued
51   FROM Accounts
52 WHERE account_type = 'savings';
53
```

The results grid displays the following data:

account_id	interest_accrued
101	260.0000
103	400.0000
105	300.0000
107	350.0000

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	17:17:13	SELECT account_id, balance * 0.05 AS interest_accrued FROM Accounts WHERE account_type = 'savings' Li...	4 row(s) returned	0.000 sec / 0.000 sec

## 11.

The screenshot shows a SQL database interface with a query editor and a results grid. The query is:

```
54  /*11*/
55 • SELECT account_id, balance
56   FROM Accounts
57 WHERE balance < -300.00;
58 /* here there was no overdraft limit given so i considered it as -300*/
59
```

The results grid displays the following data:

account_id	balance
NULL	NULL

The output pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	17:18:08	SELECT account_id, balance FROM Accounts WHERE balance < -300.00 LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec

## 12.

The screenshot shows the Oracle SQL Developer interface. The top navigation bar includes tabs like 'task 2\*', 'SQL', 'Script', 'SQL Editor', 'SQL Worksheet', 'SQL Builder', and 'SQL Tuning Advisor'. There are also icons for file operations, search, and help. A toolbar below has icons for copy, paste, find, and execute.

The main area displays a code editor with the following SQL query:

```
59
60  /*12*/
61 •  SELECT first_name, last_name
62  FROM Customers
63  WHERE address <> 'Mumbai';
64
```

Below the code is a 'Result Grid' showing the output of the query:

first_name	last_name
Rahul	Sharma
Amit	Singh
Deepa	Shah
Rajesh	Kumar
Anita	Gupta
Preeti	Agarwal

The 'Output' tab at the bottom shows the executed query and its results:

#	Time	Action
1	17:18:51	SELECT first_name, last_name FROM Customers WHERE address <> 'Mumbai' LIMIT 0, 1000

Message: 8 row(s) returned  
Duration / Fetch: 0.000 sec / 0.000 sec

## TASK3:

### 1.

The screenshot shows a SQL editor window titled "task3". The code entered is:

```
1  /* task 3*/
2
3  /*1*/
4 • SELECT AVG(balance) AS average_balance
5   FROM Accounts;
6
```

The results grid shows one row with the column "average\_balance" containing the value "11020.00000".

The output pane shows the query: "SELECT AVG(balance) AS average\_balance FROM Accounts LIMIT 0, 1000" and the message: "1 row(s) returned".

### 2.

The screenshot shows a SQL editor window titled "task3". The code entered is:

```
7  /*2*/
8 • SELECT account_id, balance
9   FROM Accounts
10  ORDER BY balance DESC
11  LIMIT 10;
12
```

The results grid shows 10 rows with columns "account\_id" and "balance". The data is:

account_id	balance
110	20000.00
108	18000.00
106	15000.00
104	12000.00
102	10000.00
109	9000.00
103	8000.00
105	7000.00
107	6000.00
101	5000.00

The output pane shows the query: "SELECT account\_id, balance FROM Accounts ORDER BY balance DESC LIMIT 10" and the message: "10 row(s) returned".

### 3.

The screenshot shows a SQL query being run in a database client. The query is:

```
13  
14  /*3*/  
15 • SELECT SUM(amount) AS total_deposits  
16  FROM Transactions  
17  WHERE transaction_type = 'deposit' AND transaction_date = '2023-01-10';  
18
```

The result grid displays one row with the value 2000.00 under the column 'total\_deposits'. The output pane shows the query was executed at 17:22:30 and returned 1 row(s) in 0.000 sec / 0.000 sec.

### 4.

The screenshot shows a SQL query being run in a database client. The query is:

```
20  
21  
22  /*4*/  
23 • SELECT MIN(DOB) AS oldest_customer, MAX(DOB) AS newest_customer  
24  FROM Customers;  
25
```

The result grid displays two rows: 'oldest\_customer' with value 1977-12-05 and 'newest\_customer' with value 1995-09-28. The output pane shows the query was executed at 17:23:06 and returned 1 row(s) in 0.000 sec / 0.000 sec.

## 5.

Screenshot of SQL Server Management Studio (SSMS) showing the results of a query. The query selects all columns from the Transactions table and joins it with the Accounts table on account\_id. The results show various transactions with their corresponding account types.

```
task* | SQL | My Snippets | SQL>Additions |
```

```
26
27  /*5*/
28 • SELECT t.*, a.account_type
29   FROM Transactions t
30   JOIN Accounts a ON t.account_id = a.account_id;
31
```

transaction_id	account_id	transaction_type	amount	transaction_date	account_type
1001	101	deposit	2000.00	2023-01-10	savings
1002	101	withdrawal	1000.00	2023-02-15	savings
1003	107	deposit	3000.00	2023-03-20	savings
1004	102	withdrawal	1500.00	2023-04-25	zero_balance
1005	104	deposit	2500.00	2023-05-10	current
1006	103	withdrawal	1200.00	2023-06-15	savings
...	...	...	...	...	...

Result 8 x

Output

#	Time	Action
1	17:23:39	SELECT t.* , a.account_type FROM Transactions t JOIN Accounts a ON t.account_id = a.account_id LIMIT 0, 1... 10 row(s) returned

Message

Duration / Fetch  
0.000 sec / 0.000 sec

## 6.

Screenshot of SQL Server Management Studio (SSMS) showing the results of a query. The query selects all columns from the Customers table and joins it with the Accounts table on customer\_id. The results show customer details and their account information.

```
task* | SQL | My Snippets | SQL>Additions |
```

```
32
33  /*6*/
34 • SELECT c.* , a.*
35   FROM Customers c
36   JOIN Accounts a ON c.customer_id = a.customer_id;
37
```

customer_id	first_name	last_name	DOB	email	phone_number	address	account_id	customer_id	account_type	balance
1	Rahul	Sharma	1990-05-15	rahul.sharma@email.com	9876543210	Delhi	101	1	savings	5200.00
2	Priya	Patel	1985-08-22	priya.patel@email.com	8765432109	Mumbai	103	2	savings	8000.00
3	Amit	Singh	1988-04-12	amit.singh@email.com	7654321098	Kolkata	104	3	current	12000.00
4	Deepa	Shah	1995-09-28	deepa.shah@email.com	6543210987	Chennai	105	4	savings	6000.00
5	Rajesh	Kumar	1977-12-05	rajesh.kumar@email.com	5432109876	Hyderabad	106	5	zero_balance	15000.00
6	Anita	Gupta	1982-03-20	anita.gupta@email.com	4321098765	Bangalore	107	6	savings	7000.00
...	...	...	...	...	...	...	...	...	...	...

Result 9 x

Output

#	Time	Action
1	17:24:15	SELECT c.* , a.* FROM Customers c JOIN Accounts a ON c.customer_id = a.customer_id LIMIT 0, 1000 10 row(s) returned

Message

Duration / Fetch  
0.000 sec / 0.000 sec

7.

The screenshot shows the SQL Server Management Studio interface with a query window titled 'task3'. The code is as follows:

```
38  /*7*/
39 •  SELECT t.*, c.*
40   FROM Transactions t
41   JOIN Accounts a ON t.account_id = a.account_id
42   JOIN Customers c ON a.customer_id = c.customer_id
43   WHERE t.account_id = '101';
44
```

The results grid displays two rows of data:

transaction_id	account_id	transaction_type	amount	transaction_date	customer_id	first_name	last_name	DOB	email	phone_number	address
1001	101	deposit	2000.00	2023-01-10	1	Rahul	Sharma	1990-05-15	rahul.sharma@email.com	9876543210	Delhi
1002	101	withdrawal	1000.00	2023-02-15	1	Rahul	Sharma	1990-05-15	rahul.sharma@email.com	9876543210	Delhi

The output pane shows the execution message:

```
1 17:25:01 SELECT t.*, c.* FROM Transactions t JOIN Accounts a ON t.account_id = a.account_id JOIN Customers c ON ... 2 row(s) returned
```

8.

The screenshot shows the SQL Server Management Studio interface with a query window titled 'task3'. The code is as follows:

```
44
45  /*8*/
46 •  SELECT customer_id, COUNT(account_id) AS num_accounts
47   FROM Accounts
48   GROUP BY customer_id
49   HAVING num_accounts > 1;
50
```

The results grid displays one row of data:

customer_id	num_accounts

The output pane shows the execution message:

```
1 17:25:31 SELECT customer_id, COUNT(account_id) AS num_accounts FROM Accounts GROUP BY customer_id HAVING num_accounts > 1; 0 row(s) returned
```

9.

task3\* x

SQLAdditions  
My Snippets

```
50
51    /*9*/
52 • SELECT account_id, SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS difference
53 FROM Transactions
54 GROUP BY account_id;
55
56
```

---

Result Grid | Filter Rows: \_\_\_\_\_ | Export: \_\_\_\_\_ | Wrap Cell Content:

account_id	difference
101	1000.00
102	-1500.00
103	-1200.00
104	6500.00
105	1700.00
107	3000.00
109	-2000.00

Result 12 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:26:06	SELECT account_id, SUM(CASE WHEN transaction_type = 'deposit' THEN amount ELSE -amount END) AS diff...	7 row(s) returned	0.000 sec / 0.000 sec

10.

task3" x

Limit to 1000 rows | My Snippets | SQLAdditions

```
56
57     /*10*/
58 • SELECT account_id, AVG(balance) AS average_daily_balance
59 FROM Accounts
60 GROUP BY account_id;
61
62
```

---

Result Grid | Filter Rows: Export: Wrap Cell Content: □

account_id	average_daily_balance
101	5200.000000
102	10000.000000
103	8000.000000
104	12000.000000
105	6000.000000
106	15000.000000
107	7000.000000
108	18000.000000

Result 14 x

Read Only | Context Help | Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	17:26:38	SELECT account_id, AVG(balance) AS average_daily_balance FROM Accounts GROUP BY account_id LIMIT ... 10 rows(s) returned		0.000 sec / 0.000 sec

## 11.

The screenshot shows a SQL database interface with a query editor and a results grid. The query is:

```
63  
64  /*11*/  
65 • SELECT account_type, SUM(balance) AS total_balance  
66  FROM Accounts  
67  GROUP BY account_type;  
68  
69
```

The results grid displays the following data:

account_type	total_balance
savings	26200.00
zero_balance	34000.00
current	50000.00

The output pane shows the execution details:

```
Result 15 x  
Output  
Action Output  
# Time Action  
1 17:27:10 SELECT account_type, SUM(balance) AS total_balance FROM Accounts GROUP BY account_type LIMIT 0, 1... 3 row(s) returned  
Duration / Fetch  
0.000 sec / 0.000 sec
```

## 12.

The screenshot shows a SQL database interface with a query editor and a results grid. The query is:

```
69  
70  /*12*/  
71 • SELECT account_id, COUNT(transaction_id) AS num_transactions  
72  FROM Transactions  
73  GROUP BY account_id  
74  ORDER BY num_transactions DESC;  
75
```

The results grid displays the following data:

account_id	num_transactions
101	2
304	2
105	2
302	1
103	1
107	1
109	1

The output pane shows the execution details:

```
Result 16 x  
Output  
Action Output  
# Time Action  
1 17:27:38 SELECT account_id, COUNT(transaction_id) AS num_transactions FROM Transactions GROUP BY account_id... 7 row(s) returned  
Duration / Fetch  
0.000 sec / 0.000 sec
```

## 13.

The screenshot shows the SQL Server Management Studio interface with a query window titled 'task3'. The query is:

```
76  /*13*/
77 • SELECT c.customer_id, c.first_name, c.last_name, a.account_type, SUM(a.balance) AS aggregate_balance
78   FROM Customers c
79   JOIN Accounts a ON c.customer_id = a.customer_id
80   GROUP BY c.customer_id, a.account_type
81   ORDER BY aggregate_balance DESC;
82
```

The results grid displays the following data:

customer_id	first_name	last_name	account_type	aggregate_balance
9	Vikram	Rajput	current	20000.00
7	Suresh	Verma	current	18000.00
5	Rajesh	Kumar	zero_balance	15000.00
3	Amit	Singh	current	12000.00
10	Neha	Sharma	zero_balance	10000.00
8	Preeti	Agarwal	zero_balance	9000.00
2	Priya	Patel	savings	8000.00
6	Anita	Gupta	savings	7000.00

The output pane shows the execution message: "1 17:28:10 SELECT c.customer\_id, c.first\_name, c.last\_name, a.account\_type, SUM(a.balance) AS aggregate\_balance FR... 10 row(s) returned".

## 14.

The screenshot shows the SQL Server Management Studio interface with a query window titled 'task3'. The query is:

```
82
83  /*14*/
84 • SELECT amount, transaction_date, account_id, COUNT(transaction_id) AS num_duplicates
85   FROM Transactions
86   GROUP BY amount, transaction_date, account_id
87   HAVING num_duplicates > 1;
88
```

The results grid displays the following data:

amount	transaction_date	account_id	num_duplicates

The output pane shows the execution message: "1 17:28:37 SELECT amount, transaction\_date, account\_id, COUNT(transaction\_id) AS num\_duplicates FROM Transactions... 0 row(s) returned".