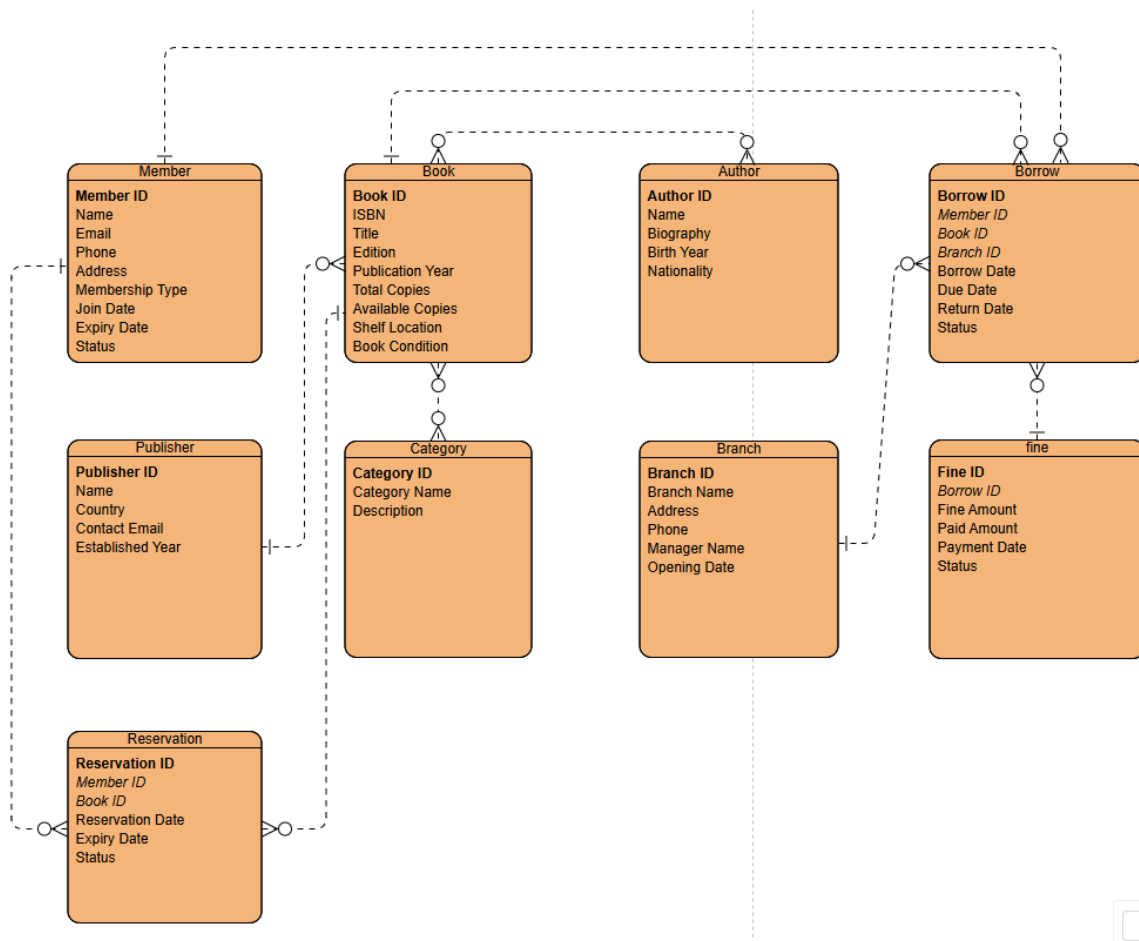


Task 1:



1. Identify normalization issues (1NF, 2NF, 3NF violations)

- 1NF:
Each column must hold only *atomic (single)* values.
No repeating columns.
Each record must be unique (use a primary key).
- 2NF:
Must already satisfy 1NF.
All non-key attributes must depend on the *whole primary key*, not just part of it.
Applies when you have a **composite key** (two or more fields as a primary key)
- 3NF:
Must already satisfy 2NF.
No non-key attribute should depend on another non-key attribute.

2. Normalize to 3NF showing each step

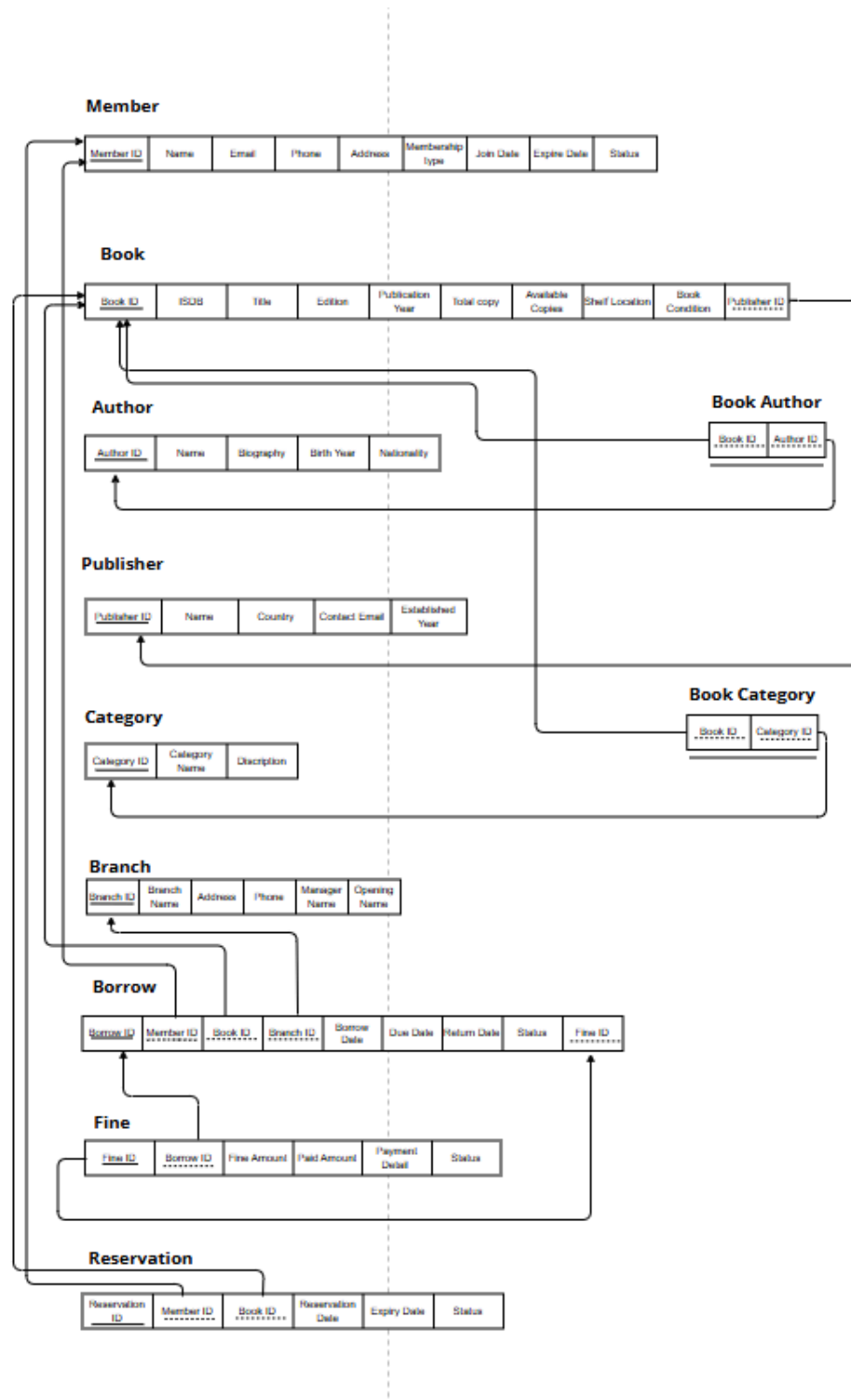
3. Explain the improvements and why each normal form matters

4. Show the final normalized tables

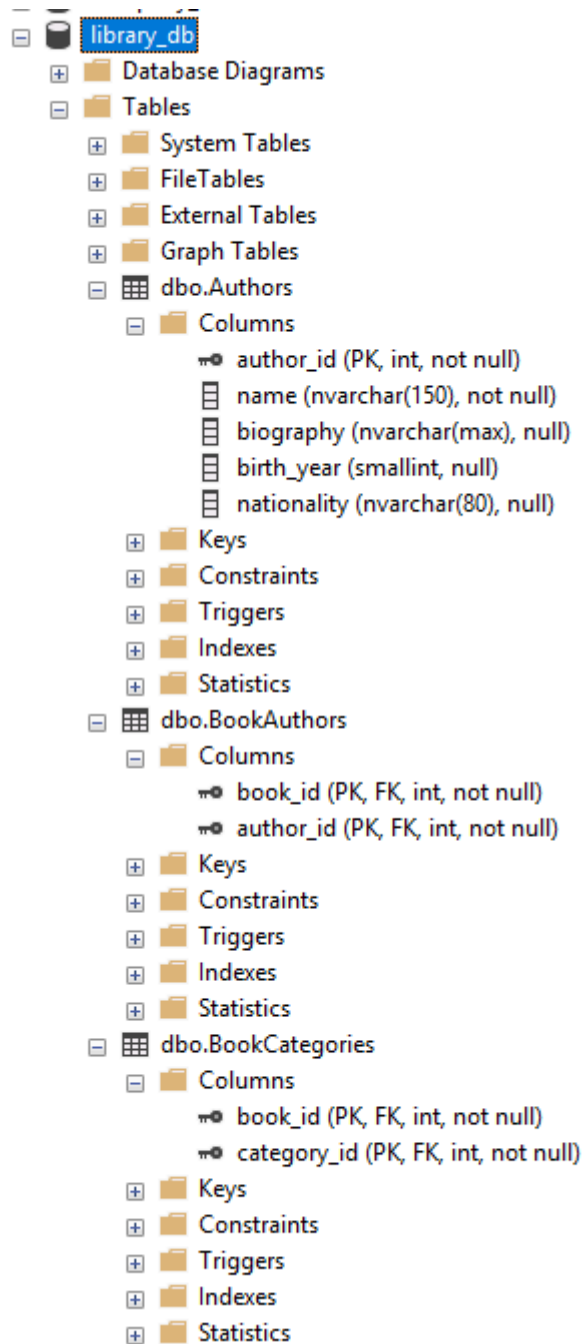
Task 3: Database Mapping

Convert your ERD into a complete relational schema:












































TASK 3:



TASK 4: SQL DDL



[-]	[Table Icon]	dbo.Books
	[-]	Columns
		book_id (PK, int, not null)
		isbn (nvarchar(20), not null)
		title (nvarchar(255), not null)
		edition (nvarchar(50), null)
		publication_year (smallint, null)
		total_copy (int, not null)
		available_copies (int, not null)
		shelf_location (nvarchar(50), null)
		book_condition (nvarchar(10), not null)
		publisher_id (FK, int, not null)
		created_at (datetime2(7), not null)
		rating (tinyint, null)
	[+]	Keys
	[+]	Constraints
	[+]	Triggers
	[+]	Indexes
	[+]	Statistics
[-]	[Table Icon]	dbo.Borrowing
	[-]	Columns
		borrow_id (PK, int, not null)
		member_id (FK, int, not null)
		book_id (FK, int, not null)
		branch_id (FK, int, not null)
		borrow_date (date, not null)
		due_date (date, not null)
		return_date (date, null)
		status (nvarchar(10), not null)
		fine_id (int, null)
	[+]	Keys
	[+]	Constraints
	[+]	Triggers
	[+]	Indexes
	[+]	Statistics
[-]	[Table Icon]	dbo.Branches
	[-]	Columns
		branch_id (PK, int, not null)
		branch_name (nvarchar(100), not null)
		address (nvarchar(255), null)
		phone (nvarchar(30), null)
		manager_name (nvarchar(120), null)
		opening_name (nvarchar(120), null)
	[+]	Keys
	[+]	Constraints

- [-]  **dbo.Categories**
 - [-]  Columns
 -  category_id (PK, int, not null)
 -  category_name (nvarchar(100), not null)
 -  description (nvarchar(255), null)
 - [+]  Keys
 - [+]  Constraints
 - [+]  Triggers
 - [+]  Indexes
 - [+]  Statistics
- [-]  **dbo.Fines**
 - [-]  Columns
 -  fine_id (PK, int, not null)
 -  borrow_id (FK, int, not null)
 -  fine_amount (decimal(8,3), not null)
 -  paid_amount (decimal(8,3), not null)
 -  payment_detail (nvarchar(255), null)
 -  status (nvarchar(20), not null)
 -  issued_date (date, not null)
 -  paid_date (date, null)
 - [+]  Keys
 - [+]  Constraints
 - [+]  Triggers
 - [+]  Indexes
 - [+]  Statistics
- [-]  **dbo.Members**
 - [-]  Columns
 -  member_id (PK, int, not null)
 -  name (nvarchar(120), not null)
 -  email (nvarchar(255), not null)
 -  phone (nvarchar(30), null)
 -  address (nvarchar(255), null)
 -  membership_type (nvarchar(10), not null)
 -  join_date (date, not null)
 -  expire_date (date, null)
 -  status (nvarchar(10), not null)
 -  created_at (datetime2(7), not null)
 -  last_login (datetime2(7), null)
 - [+]  Keys
 - [+]  Constraints
 - [+]  Triggers
 - [+]  Indexes
 - [+]  Statistics

- Statistics
- dbo.Publishers**
 - Columns
 - publisher_id (PK, int, not null)
 - name (nvarchar(150), not null)
 - country (nvarchar(80), null)
 - contact_email (nvarchar(255), null)
 - established_year (smallint, null)
 - Keys
 - Constraints
 - Triggers
 - Indexes
 - Statistics
- dbo.Publishers_Addresses**
 - Columns
 - publisher_id (PK, FK, int, not null)
 - Keys
 - Constraints
 - Triggers
 - Indexes
 - Statistics
- dbo.Reservations**
 - Columns
 - reservation_id (PK, int, not null)
 - member_id (FK, int, not null)
 - book_id (FK, int, not null)
 - reservation_date (date, not null)
 - expiry_date (date, null)
 - status (nvarchar(10), not null)
 - Keys
 - Constraints
 - Triggers
 - Indexes
 - Statistics

TASK 5:

Results Messages					
	author_id	name	biography	birth_year	nationality
1	1	Naguib Mahfouz	Egyptian novelist	1911	Egyptian
2	2	Chinua Achebe	Nigerian novelist	1930	Nigerian
3	3	J. K. Rowling	British author	1965	British
4	4	George Orwell	English writer	1903	British
5	5	Haruki Murakami	Japanese novelist	1949	Japanese
6	6	Agatha Christie	Mystery writer	1890	British
7	7	Yuval Noah Harari	Historian	1976	Israeli
8	8	Isabel Allende	Chilean writer	1942	Chilean
9	9	Paulo Coelho	Brazilian writer	1947	Brazilian
10	10	Khaled Hosseini	Afghan-American	1965	American

Results Messages		
	book_id	author_id
1	8	3
2	9	3
3	2	4
4	3	4
5	27	4
6	4	5
7	5	5
8	6	6
9	7	6
10	13	7
11	14	7
12	21	8
13	22	8
14	1	9
15	10	10
16	11	10

Results Messages									
	borrow_id	member_id	book_id	branch_id	borrow_date	due_date	return_date	status	fine_id
1	1	2	2	1	2025-01-05	2025-01-20	NULL	Borrowed	NULL
2	2	1	1	2	2025-01-10	2025-01-25	2025-01-22	Returned	NULL
3	3	4	6	3	2025-01-12	2025-01-27	NULL	Overdue	1
4	4	5	8	4	2025-01-01	2025-01-16	2025-01-15	Returned	NULL
5	5	6	9	1	2025-02-01	2025-02-16	NULL	Borrowed	6
6	6	7	13	5	2025-02-05	2025-02-20	NULL	Borrowed	NULL
7	7	8	14	5	2024-12-10	2024-12-25	2024-12-28	Returned	NULL
8	8	9	17	1	2025-02-06	2025-02-21	NULL	Borrowed	7
9	9	10	18	2	2025-01-15	2025-01-30	2025-01-27	Returned	NULL
10	10	11	19	3	2025-02-10	2025-02-25	NULL	Borrowed	NULL
11	11	12	20	4	2024-11-01	2024-11-16	NULL	Overdue	2
12	12	13	21	1	2025-02-12	2025-02-27	NULL	Borrowed	NULL
13	13	14	22	5	2024-10-10	2024-10-25	2024-10-22	Returned	NULL
14	14	15	23	3	2025-02-01	2025-02-16	NULL	Borrowed	NULL
15	15	16	24	4	2025-01-22	2025-02-06	2025-02-03	Returned	NULL
16	16	17	25	2	2024-12-01	2024-12-16	NULL	Overdue	3
17	17	18	26	1	2025-02-03	2025-02-18	NULL	Borrowed	8
18	18	19	27	5	2024-11-20	2024-12-05	NULL	Overdue	4
19	19	20	28	5	2025-02-14	2025-03-01	NULL	Borrowed	NULL

Results Messages						
	branch_id	branch_name	address	phone	manager_name	opening_name
1	1	Ruwi Central	Ruwi, Muscat	+968-2412-0001	Aisha Al-Maskari	Grand Opening
2	2	Qurum Heights	Qurum, Muscat	+968-2412-0002	Hamood Al-Harthy	Qurum Fest
3	3	Al Khoud	Al Khoud, Seeb	+968-2412-0003	Latifa Al-Farsi	Seeb Reads
4	4	Sohar Comiche	Comiche Rd, Sohar	+968-2684-0001	Rashid Al-Balushi	Sea & Books
5	5	Nizwa Fort	Fort Area, Nizwa	+968-2540-0001	Maha Al-Busaidi	Heritage Week

Results Messages			
	category_id	category_name	description
1	1	Fiction	General fiction
2	2	Non-Fiction	Essays & facts
3	3	History	Historical works
4	4	Science	Science & tech
5	5	Fantasy	Magic & fantasy
6	6	Mystery	Detective & crime
7	7	Philosophy	Philosophical
8	8	Biography	Life stories

Results Messages								
	fine_id	borrow_id	fine_amount	paid_amount	payment_detail	status	issued_date	paid_date
1	1	3	3.000	0.000	Overdue 5 days	Unpaid	2025-01-28	NULL
2	2	11	5.000	5.000	Late fee paid cash	Paid	2024-11-20	2024-11-22
3	3	16	4.000	0.000	Overdue notice	Unpaid	2024-12-20	NULL
4	4	18	6.000	2.000	Partial payment	Partially Paid	2024-12-10	NULL
5	5	23	4.000	0.000	Overdue 3 days	Unpaid	2025-01-22	NULL
6	6	5	2.000	2.000	Late fee paid online	Paid	2025-02-20	2025-02-21
7	7	8	3.000	0.000	Pending	Unpaid	2025-02-25	NULL
8	8	17	1.500	1.500	Paid at desk	Paid	2025-02-18	2025-02-18

Results Messages											
	member_id	name	email	phone	address	membership_type	join_date	expire_date	status	created_at	last_login
1	1	Mohammed Anad	mohammed1@example.com	94958181	Muscat	Premium	2023-02-10	2026-02-10	Active	2025-11-04 10:41:12.7948840	NULL
2	2	Sara Al-Harthy	sara2@example.com	92111112	Muscat	Standard	2024-04-01	2025-04-01	Active	2025-11-04 10:41:12.7948840	NULL
3	3	Yousef Al-Fansi	yousef3@example.com	92222223	Seeb	Standard	2022-06-15	2023-06-15	Expired	2025-11-04 10:41:12.7948840	NULL
4	4	Fatma Al-Balushi	fatma4@example.com	93333334	Sohar	Premium	2024-09-12	2026-09-12	Active	2025-11-04 10:41:12.7948840	NULL
5	5	Huda Al-Maskari	huda5@example.com	94444445	Nizwa	Standard	2021-01-10	2022-01-10	Expired	2025-11-04 10:41:12.7948840	NULL
6	6	Khalid Al-Hina	khalid6@example.com	95555556	Muscat	Standard	2025-01-02	2026-01-02	Active	2025-11-04 10:41:12.7948840	NULL
7	7	Maha Al-Busaidi	maha7@example.com	96666667	Qurum	Premium	2024-02-20	2026-02-20	Active	2025-11-04 10:41:12.7948840	NULL
8	8	Salim Al-Abri	salim8@example.com	97777778	Ibra	Standard	2023-05-05	2024-05-05	Expired	2025-11-04 10:41:12.7948840	NULL
9	9	Noor Al-Lawati	noor9@example.com	98888889	Barka	Premium	2024-07-18	2026-07-18	Active	2025-11-04 10:41:12.7948840	NULL
10	10	Ahmed Al-Hashmi	ahmed10@example.com	99999990	Muscat	Standard	2025-02-01	2026-02-01	Active	2025-11-04 10:41:12.7948840	NULL
11	11	Zainab Al-Said	zainab11@example.com	90111112	Seeb	Standard	2023-09-09	2025-09-09	Active	2025-11-04 10:41:12.7948840	NULL
12	12	Faisal Al-Kalbani	faisal12@example.com	90222223	Musannah	Standard	2022-03-01	2023-03-01	Expired	2025-11-04 10:41:12.7948840	NULL
13	13	Maryam Al-Azri	maryam13@example.com	90333334	Salalah	Premium	2024-11-11	2026-11-11	Active	2025-11-04 10:41:12.7948840	NULL
14	14	Hassan Al-Raisi	hassan14@example.com	90444445	Sohar	Standard	2024-01-01	2025-01-01	Expired	2025-11-04 10:41:12.7948840	NULL
15	15	Alya Al-Mahizi	alya15@example.com	90555556	Muscat	Premium	2025-01-15	2027-01-15	Active	2025-11-04 10:41:12.7948840	NULL
16	16	Imran Al-Badri	imran16@example.com	90666667	Ibri	Standard	2025-03-03	2026-03-03	Active	2025-11-04 10:41:12.7948840	NULL
17	17	Rania Al-Habsi	rania17@example.com	90777778	Qurayat	Standard	2023-12-12	2024-12-12	Expired	2025-11-04 10:41:12.7948840	NULL
18	18	Adil Al-Sinani	adil18@example.com	90888889	Seeb	Premium	2025-01-25	2027-01-25	Active	2025-11-04 10:41:12.7948840	NULL
19	19	Layla Al-Harthy	layla19@example.com	90999990	Rustaq	Standard	2024-06-06	2025-06-06	Expired	2025-11-04 10:41:12.7948840	NULL

Results Messages					
	publisher_id	name	country	contact_email	established_year
1	1	Penguin Random House	UK	contact@prh.com	1935
2	2	HarperCollins	USA	info@harper.com	1989
3	3	Oxford Univ. Press	UK	oup@oup.com	1586
4	4	Bloomsbury	UK	hello@blooms.com	1986
5	5	Al Shabiba Press	Oman	pub@shabiba.om	1992

	reservation_id	member_id	book_id	reservation_date	expiry_date	status
1	2	2	17	2025-01-20	2025-01-27	Cancelled
2	3	15	21	2025-02-10	2025-02-17	Fulfilled

Task 6: SQL DQL - Basic Queries

1- List all books published after 2020

```
SELECT *
FROM Books
WHERE publication_year > 2020
```

	book_id	isbn	title	edition	publication_year	total_copy	available_copies	shelf_location	book_condition	publisher_id	created_at	rating
1	4	9780000000004	Kafka on the Shore	1st	2002	5	3	F1-B1	Good	4	2025-11-04 10:41:12.8616444	5
2	10	9780000000010	Kite Runner	1st	2003	8	6	F1-C1	Good	1	2025-11-04 10:41:12.8616444	4
3	11	9780000000011	A Thousand Splendid Suns	1st	2007	8	6	F1-C2	Good	1	2025-11-04 10:41:12.8616444	5
4	13	9780000000013	Sapiens	1st	2011	10	7	NF1-A1	Good	2	2025-11-04 10:41:12.8616444	5
5	14	9780000000014	Homo Deus	1st	2016	10	8	NF1-A2	Good	2	2025-11-04 10:41:12.8616444	4
6	16	9780000000016	Lean In	1st	2013	6	5	NF1-A3	Good	1	2025-11-04 10:41:12.8616444	4
7	25	9780000000025	Thinking, Fast and Slow	1st	2011	9	7	NF1-B1	Good	1	2025-11-04 10:41:12.8616444	5
8	26	9780000000026	Blink	1st	2005	8	6	NF1-B2	Good	1	2025-11-04 10:41:12.8616444	4
9	30	9780000000030	The Subtle Art of Not Giving a F*ck	1st	2016	9	6	NF1-B3	Good	2	2025-11-04 10:41:12.8616444	4

2-Find all members whose membership expires in the next 30 days

```
SELECT *FROM dbo.Members
WHERE expire_date
BETWEEN CAST(GETDATE() AS date)
AND DATEADD(day, 30, CAST(GETDATE() AS date));
```

	member_id	name	email	phone	address	membership_type	join_date	expire_date	status	created_at	last_login
--	-----------	------	-------	-------	---------	-----------------	-----------	-------------	--------	------------	------------

3-Show all overdue borrowings (return_date IS NULL AND due_date < CURDATE())

```
SELECT *
FROM Borrowing br
JOIN Members m ON m.member_id = br.member_id
JOIN Books b ON b.book_id = br.book_id
WHERE br.return_date IS NULL
AND br.due_date < CAST(GETDATE() AS date);
```

00 %

Results Messages

	book_id	member_id	branch_id	borrow_date	due_date	return_date	status	fine_id	member_id	name	email	phone	address	membership_type	join_date	expire_date	status	created_at	
1	3	2	1	2025-01-05	2025-01-20	NULL	Borrowed	NULL	2	Sara Al-Hathry	sara24@example.com	92111112	Muscat	Standard	2024-04-01	2025-04-01	Active	2025-11-04 10:41:12.7948840	
2	4	6	3	2025-01-12	2025-01-27	NULL	Overdue	1	4	Fatma Al-Balushi	fatma4@example.com	93333334	Sohar	Premium	2024-09-12	2026-09-12	Active	2025-11-04 10:41:12.7948840	
3	5	6	9	1	2025-02-01	2025-02-16	NULL	Borrowed	6	6	Khalid Al-Hina	khalid5@example.com	95555556	Muscat	Standard	2025-01-02	2026-01-02	Active	2025-11-04 10:41:12.7948840
4	6	7	13	5	2025-02-05	2025-02-20	NULL	Borrowed	NULL	7	Maha Al-Busaidi	maha7@example.com	96666667	Qurum	Premium	2024-02-20	2026-02-20	Active	2025-11-04 10:41:12.7948840
5	8	9	17	1	2025-02-06	2025-02-21	NULL	Borrowed	7	9	Noor Al-Lawati	noor9@example.com	98888889	Barka	Premium	2024-07-18	2026-07-18	Active	2025-11-04 10:41:12.7948840
6	10	11	19	3	2025-02-10	2025-02-25	NULL	Borrowed	NULL	11	Zainab Al-Said	zainab11@example.com	90111112	Seeb	Standard	2023-09-09	2025-09-09	Active	2025-11-04 10:41:12.7948840
7	11	12	20	4	2024-11-01	2024-11-16	NULL	Overdue	2	12	Faisal Al-Kalbani	faisal12@example.com	90222223	Musannah	Standard	2022-03-01	2023-03-01	Expired	2025-11-04 10:41:12.7948840
8	12	13	21	1	2025-02-12	2025-02-27	NULL	Borrowed	NULL	13	Mayam Al-Azri	mayam13@example.com	90333334	Salalah	Premium	2024-11-11	2026-11-11	Active	2025-11-04 10:41:12.7948840
9	14	15	23	3	2025-02-01	2025-02-16	NULL	Borrowed	NULL	15	Alya Al-Mahitzi	alya15@example.com	90555556	Muscat	Premium	2025-01-15	2027-01-15	Active	2025-11-04 10:41:12.7948840
10	16	17	25	2	2024-12-01	2024-12-16	NULL	Overdue	3	17	Rania Al-Habshi	rania17@example.com	90777778	Quryat	Standard	2023-12-12	2024-12-12	Expired	2025-11-04 10:41:12.7948840
11	17	18	26	1	2025-02-03	2025-02-18	NULL	Borrowed	8	18	Adil Al-Sinani	adil18@example.com	90888889	Seeb	Premium	2025-01-25	2027-01-25	Active	2025-11-04 10:41:12.7948840
12	18	19	27	5	2024-11-20	2024-12-05	NULL	Overdue	4	19	Layla Al-Hathry	layla19@example.com	90999990	Rustaq	Standard	2024-06-06	2025-06-06	Expired	2025-11-04 10:41:12.7948840
13	19	20	28	5	2025-02-14	2025-03-01	NULL	Borrowed	NULL	20	Waleed Al-Khndi	waleed20@example.com	91010101	Muscat	Standard	2025-02-20	2026-02-20	Active	2025-11-04 10:41:12.7948840
14	21	1	30	4	2025-02-15	2025-03-02	NULL	Borrowed	NULL	1	Mohammed Anad	mohammed1@example.com	94958181	Muscat	Premium	2023-02-10	2026-02-10	Active	2025-11-04 10:41:12.7948840
15	23	3	11	3	2025-01-03	2025-01-18	NULL	Overdue	5	3	Yousef Al-Farsi	yousef3@example.com	92222223	Seeb	Standard	2022-06-15	2023-06-15	Expired	2025-11-04 10:41:12.7948840
16	24	9	12	4	2025-02-10	2025-02-25	NULL	Borrowed	NULL	9	Noor Al-Lawati	noor9@example.com	98888889	Barka	Premium	2024-07-18	2026-07-18	Active	2025-11-04 10:41:12.7948840

4. List books that have never been borrowed

```
SELECT b.book_id, b.title
FROM Books b
LEFT JOIN Borrowing br ON br.book_id = b.book_id
WHERE br.borrow_id IS NULL;
```

100 %

Results Messages

	book_id	title
1	3	Animal Farm
2	4	Kafka on the Shore
3	16	Lean In
4	7	Murder on the Orient Express
5	5	Norwegian Wood

5. Find members with pending fines greater than \$3

```
SELECT m.member_id, m.name, f.fine_amount, f.paid_amount,  
       f.fine_amount - f.paid_amount AS pending_amount  
FROM Fines f  
JOIN Borrowing br ON br.borrow_id = f.borrow_id  
JOIN Members m ON m.member_id = br.member_id  
WHERE (f.fine_amount - f.paid_amount) > 3  
       AND f.status IN (N'Unpaid', N'Partially Paid');
```

100 %

	member_id	name	fine_amount	paid_amount	pending_amount
1	17	Rania Al-Habsi	4.000	0.000	4.000
2	19	Layla Al-Harthy	6.000	2.000	4.000
3	3	Yousef Al-Farsi	4.000	0.000	4.000

4- Show all books in 'Fiction' category

SQLQuery4.sql - DE...6518\CodeLine (57)* SQLQuery3.sql - DE...6518\CodeLine (6

```
SELECT b.book_id, b.title  
FROM Books b  
JOIN BookCategories bc ON bc.book_id = b.book_id  
JOIN Categories c ON c.category_id = bc.category_id  
WHERE c.category_name = N'Fiction';
```

100 %

	book_id	title
1	1	The Alchemist
2	2	1984
3	3	Animal Farm
4	4	Kafka on the Shore

7. Display books with less than 2 available copies

```
SELECT *
FROM Books
WHERE available_copies < 2;
```

100 %

Results Messages

	book_id	isbn	title	edition	publication_year	total_copy	available_copies	shelf_location	book_condition	publisher_id	created_at	rating
1	3	978000000003	Animal Farm	1st	1945	6	1	F1-A3	Fair	1	2025-11-04 10:41:12.8616444	4

8. List all authors from 'USA' or 'UK'

SQLQuery4.sql - DE...b518\CodeLine (57))

```
SELECT author_id, name, nationality
FROM Authors
WHERE nationality IN ( N'American', N'British');
```

100 %

Results Messages

	author_id	name	nationality
1	3	J. K. Rowling	British
2	4	George Orwell	British
3	6	Agatha Christie	British
4	10	Khaled Hosseini	American

9. Find books published by 'Penguin Random House'

SQLQuery4.sql - DE...6518\CodeLine (37))

```
SELECT b.book_id, b.title
FROM dbo.Books b
JOIN dbo.Publishers p ON p.publisher_id = b.publisher_id
WHERE p.name = N'Penguin Random House';
```

100 %

Results Messages

	book_id	title
1	2	1984
2	3	Animal Farm
3	10	Kite Runner
4	11	A Thousand Splendid Suns
5	16	Lean In
6	19	The Stranger
7	21	Love in the Time of Cholera
8	23	The Old Man and the Sea
9	25	Thinking, Fast and Slow
10	26	Blink

10. Show borrowings from last month using BETWEEN

SQLQuery4.sql - DE...6518\CodeLine (57))

```
DECLARE @start_last_month date = DATEFROMPARTS(YEAR(DATEADD(month,-1, GETDATE())),
MONTH(DATEADD(month,-1, GETDATE())), 1);
DECLARE @end_last_month date = EOMONTH(DATEADD(month,-1, GETDATE()));

SELECT br.borrow_id, m.name AS member_name, b.title, br.borrow_date, br.due_date, br.status
FROM dbo.Borrowing br
JOIN dbo.Members m ON m.member_id = br.member_id
JOIN dbo.Books b ON b.book_id = br.book_id
WHERE br.borrow_date BETWEEN @start_last_month AND @end_last_month
ORDER BY br.borrow_date;
```

100 %

Results Messages

	borrow_id	member_name	title	borrow_date	due_date	status
1	26	Waleed Al-Kindi	Blink	2025-10-06	2025-10-20	Borrowed
2	27	Adil Al-Sinani	The Subtle Art of Not Giving a F*ck	2025-10-13	2025-10-27	Borrowed
3	28	Maryam Al-Azri	Brief History of Time	2025-10-22	2025-11-05	Borrowed

Task 8: Aggregation Functions

1. COUNT:

```
SELECT m.member_id, m.name, COUNT(*) AS borrow_count
FROM Borrowing br
JOIN Members m ON m.member_id = br.member_id
GROUP BY m.member_id, m.name
ORDER BY borrow_count DESC;
```

100 %

Results Messages

	member_id	name	borrow_count
1	1	Mohammed Anad	2
2	2	Sara Al-Harthy	2
3	5	Huda Al-Maskari	2
4	9	Noor Al-Lawati	2
5	10	Ahmed Al-Hashmi	2
6	13	Maryam Al-Azri	2
7	18	Adil Al-Sinani	2
8	20	Waleed Al-Kindi	2
9	19	Layla Al-Harthy	1
10	14	Hassan Al-Raisi	1
11	15	Alya Al-Mahrizi	1
12	16	Imran Al-Badri	1
13	17	Rania Al-Habsi	1
14	11	Zainab Al-Said	1
15	12	Faisal Al-Kalbani	1
16	6	Khalid Al-Hina	1
17	7	Maha Al-Busaidi	1
18	8	Salim Al-Abri	1
19	3	Yousef Al-Farsi	1

SQLQuery4.sql - DE...65f8\CodeLine (57))* X SQLQuery3.sql - DE...65f8\CodeLine (60))* SQLQuer

```

SELECT c.category_id, c.category_name, COUNT(bc.book_id) AS books_in_category
FROM Categories c
LEFT JOIN BookCategories bc ON bc.category_id = c.category_id
GROUP BY c.category_id, c.category_name
ORDER BY books_in_category DESC;

```

100 %

Results Messages

	category_id	category_name	books_in_category
1	1	Fiction	4
2	2	Non-Fiction	3
3	4	Science	2
4	5	Fantasy	2
5	6	Mystery	2
6	7	Philosophy	2
7	8	Biography	1
8	3	History	0

SQLQuery4.sql - DE...6518\CodeLine (57))* X SQLQuery3.sql - DE...65

```
SELECT bch.branch_id, bch.branch_name,  
       COUNT (*) AS overdue_count  
FROM Borrowing br  
JOIN Branches bch ON bch.branch_id = br.branch_id  
WHERE br.return_date IS NULL  
      AND br.due_date < CAST(GETDATE() AS date)  
GROUP BY bch.branch_id, bch.branch_name  
ORDER BY overdue_count DESC;
```

100 %

Results Messages

	branch_id	branch_name	overdue_count
1	1	Ruwi Central	6
2	3	Al Khoud	4
3	4	Sohar Comiche	3
4	5	Nizwa Fort	3
5	2	Qurum Heights	2

SUM

```
SELECT DATEFROMPARTS(YEAR(ISNULL(paid_date, issued_date)),  
                     MONTH(ISNULL(paid_date, issued_date)), 1) AS month_start,  
       SUM(paid_amount) AS total_collected  
FROM Fines  
GROUP BY DATEFROMPARTS(YEAR(ISNULL(paid_date, issued_date)),  
                       MONTH(ISNULL(paid_date, issued_date)), 1)  
ORDER BY month_start;
```

100 %

Results Messages

	month_start	total_collected
1	2024-11-01	5.000
2	2024-12-01	2.000
3	2025-01-01	0.000
4	2025-02-01	3.500

```

SELECT m.member_id, m.name,
       SUM(CASE WHEN f.fine_amount - f.paid_amount > 0
                 THEN f.fine_amount - f.paid_amount ELSE 0 END) AS pending_total
FROM Members m
JOIN Borrowing br ON br.member_id = m.member_id
JOIN Fines f      ON f.borrow_id = br.borrow_id
GROUP BY m.member_id, m.name
HAVING SUM(CASE WHEN f.fine_amount - f.paid_amount > 0
                 THEN f.fine_amount - f.paid_amount ELSE 0 END) > 0
ORDER BY pending_total DESC;

```

100 %

Results Messages

	member_id	name	pending_total
1	3	Yousef Al-Farsi	4.000
2	17	Rania Al-Habsi	4.000
3	19	Layla Al-Harthy	4.000
4	4	Fatma Al-Balushi	3.000
5	9	Noor Al-Lawati	3.000

SQLQuery4.sql - DE...65I8\CodeLine (57))* -X SQLQuery3.sql - DE...65I8\Cod

```
SELECT p.publisher_id, p.name AS publisher_name,  
       SUM(b.available_copies) AS total_available_copies  
FROM Publishers p  
JOIN Books b ON b.publisher_id = p.publisher_id  
GROUP BY p.publisher_id, p.name  
ORDER BY total_available_copies DESC;
```

100 %

Results Messages

	publisher_id	publisher_name	total_available_copies
1	1	Penguin Random House	42
2	3	Oxford Univ. Press	40
3	2	HarperCollins	34
4	4	Bloomsbury	25
5	5	Al Shabiba Press	4

AVG:

```
SQLQuery4.sql - DE...65I8\CodeLine (57))* X SQLQuery3.sql - DE...65I8\CodeLine (60))* SQLQuery2.sql
SELECT AVG(DATEDIFF(day, borrow_date, return_date)*1.0) AS avg_days_borrowed
FROM Borrowing
WHERE return_date IS NOT NULL;
```

100 %

Results Messages

	avg_days_borrowed
1	12.333333

SQLQuery4.sql - DE...6518\CodeLine (57))* X SQLQuery3.sql - DE...6518

```
SELECT m.member_id, m.name,  
       AVG(f.fine_amount*1.0) AS avg_fine_amount  
FROM Members m  
JOIN Borrowing br ON br.member_id = m.member_id  
JOIN Fines f      ON f.borrow_id = br.borrow_id  
GROUP BY m.member_id, m.name  
ORDER BY avg_fine_amount DESC;
```

100 %

Results Messages

	member_id	name	avg_fine_amount
1	19	Layla Al-Harthy	6.000000
2	12	Faisal Al-Kalbani	5.000000
3	17	Rania Al-Habsi	4.000000
4	3	Yousef Al-Farsi	4.000000
5	4	Fatma Al-Balushi	3.000000
6	9	Noor Al-Lawati	3.000000
7	6	Khalid Al-Hina	2.000000
8	18	Adil Al-Sinani	1.500000

SQLQuery4.sql - DE...6518\CodeLine (57))* X SQLQuery3.sql - DE...6518\Code

```
WITH per_branch AS (  
    SELECT br.branch_id, COUNT(*) AS borrow_count  
    FROM Borrowing br  
    GROUP BY br.branch_id  
)  
SELECT AVG(borrow_count*1.0) AS avg_borrows_per_branch  
FROM per_branch;
```

100 %

Results Messages

	avg_borrows_per_branch
1	5.600000

4) MIN / MAX

```
SELECT MIN(publication_year) AS oldest_year,  
       MAX(publication_year) AS newest_year  
FROM Books;
```

100 %

Results Messages

	oldest_year	newest_year
1	180	2016

SQLQuery4.sql - DE...6518\CodeLine (57))* SQLQuery3.sql - DE...6518\CodeLine (60))*

```

SELECT TOP 1 WITH TIES m.member_id, m.name, COUNT(*) AS borrow_count
FROM Borrowing br
JOIN Members m ON m.member_id = br.member_id
GROUP BY m.member_id, m.name
ORDER BY borrow_count DESC;

SELECT TOP 1 WITH TIES m.member_id, m.name, COUNT(*) AS borrow_count
FROM Borrowing br
JOIN Members m ON m.member_id = br.member_id
GROUP BY m.member_id, m.name
ORDER BY borrow_count ASC;

```

100 %

Results Messages

	member_id	name	borrow_count
1	1	Mohammed Anad	2
2	2	Sara Al-Harthy	2
3	5	Huda Al-Maskari	2
4	9	Noor Al-Lawati	2
5	10	Ahmed Al-Hashmi	2
6	13	Maryam Al-Azri	2
7	18	Adil Al-Sinani	2
8	20	Waleed Al-Kindi	2

	member_id	name	borrow_count
1	3	Yousef Al-Farsi	1
2	4	Fatma Al-Balushi	1
3	6	Khalid Al-Hina	1
4	7	Maha Al-Busaidi	1
5	8	Salim Al-Abri	1
6	14	Hassan Al-Raisi	1
7	15	Alya Al-Mahrizi	1
8	16	Imran Al-Badri	1
9	17	Rania Al-Habsi	1

SQLQuery4.sql - DE...65I8\CodeLine (57))* X SQLQuery3.sql - DE...65I8\CodeLine (60))* SC

```
SELECT TOP 1 WITH TIES f.fine_id, m.name, f.fine_amount, f.issued_date  
FROM Fines f  
JOIN Borrowing br ON br.borrow_id = f.borrow_id  
JOIN Members m ON m.member_id = br.member_id  
ORDER BY f.fine_amount DESC;
```

100 %

Results Messages

	fine_id	name	fine_amount	issued_date
1	4	Layla Al-Harthy	6.000	2024-12-10

Complex Aggregations with HAVING

```
SQLQuery4.sql - DE...0518(CodeLine (37))  SQLQuery5.sql - DE...0518(CodeLine (66))
SELECT c.category_id, c.category_name, COUNT(bc.book_id) AS book_count
FROM Categories c
JOIN BookCategories bc ON bc.category_id = c.category_id
GROUP BY c.category_id, c.category_name
HAVING COUNT(DISTINCT bc.book_id) > 3
ORDER BY book_count DESC;
```

100 %

Results Messages

	category_id	category_name	book_count
1	1	Fiction	4

SQLQuery4.sql - DE...6518\CodeLine (57))* SQLQuery3.sql - DE...6518\CodeLine (60))* SQLQuery2.sql - DE...

```

DECLARE @start_last_month date = DATEFROMPARTS(YEAR(DATEADD(month,-1, GETDATE())),
                                                MONTH(DATEADD(month,-1, GETDATE())), 1);
DECLARE @end_last_month    date = EOMONTH(DATEADD(month,-1, GETDATE()));

SELECT m.member_id, m.name, COUNT(*) AS borrows_last_month
FROM Borrowing br
JOIN Members m ON m.member_id = br.member_id
WHERE br.borrow_date BETWEEN @start_last_month AND @end_last_month
GROUP BY m.member_id, m.name
HAVING COUNT(*) > 3
ORDER BY borrows_last_month DESC;

```

100 %

Results Messages

member_id	name	borrows_last_month
-----------	------	--------------------

SQLQuery4.sql - DE...6518\CodeLine (57))* SQLQuery3.sql - DE...6518\CodeLine (60))* SQLQuery2.sql - DE...

```

SELECT a.author_id, a.name, COUNT(DISTINCT ba.book_id) AS book_count
FROM Authors a
JOIN BookAuthors ba ON ba.author_id = a.author_id
GROUP BY a.author_id, a.name
HAVING COUNT(DISTINCT ba.book_id) > 2
ORDER BY book_count DESC;

```

100 %

Results Messages

	author_id	name	book_count
1	4	George Orwell	3

6) GROUP BY with Multiple Columns

SQLQuery4.sql - DE...65I8\CodeLine (57))* SQLQuery3.sql - DE...65I8\CodeLine (60))* SQLQuery2.sql - DE...65

```
SELECT bch.branch_id, bch.branch_name,  
       DATEFROMPARTS(YEAR(br.borrow_date), MONTH(br.borrow_date), 1) AS month_start,  
       COUNT(*) AS borrow_count  
FROM Borrowing br  
JOIN Branches bch ON bch.branch_id = br.branch_id  
GROUP BY bch.branch_id, bch.branch_name,  
         DATEFROMPARTS(YEAR(br.borrow_date), MONTH(br.borrow_date), 1)  
ORDER BY bch.branch_name, month_start;
```

100 %

Results Messages

	branch_id	branch_name	month_start	borrow_count
1	3	Al Khoud	2025-01-01	2
2	3	Al Khoud	2025-02-01	2
3	3	Al Khoud	2025-10-01	1
4	5	Nizwa Fort	2024-10-01	1
5	5	Nizwa Fort	2024-11-01	1
6	5	Nizwa Fort	2024-12-01	1
7	5	Nizwa Fort	2025-02-01	2
8	2	Qurum Heights	2024-12-01	1
9	2	Qurum Heights	2025-01-01	3
10	2	Qurum Heights	2025-10-01	1
11	1	Ruwi Central	2025-01-01	3
12	1	Ruwi Central	2025-02-01	4
13	1	Ruwi Central	2025-10-01	1
14	4	Sohar Comiche	2024-11-01	1
15	4	Sohar Comiche	2025-01-01	2
16	4	Sohar Comiche	2025-02-01	2

7) Subqueries with Aggregation

SQLQuery4.sql - DE...6518\CodeLine (57))* -p X SQLQuery3.sql - DE...6518\CodeLine (60))*

```
WITH per_book AS (  
    SELECT br.book_id, COUNT(*) AS borrow_cnt  
    FROM Borrowing br  
    GROUP BY br.book_id  
) , avg_cnt AS (  
    SELECT AVG(borrow_cnt*1.0) AS avg_borrows FROM per_book  
)  
SELECT b.book_id, b.title, p.borrow_cnt  
FROM per_book p  
CROSS JOIN avg_cnt a  
JOIN Books b ON b.book_id = p.book_id  
WHERE p.borrow_cnt > a.avg_borrows  
ORDER BY p.borrow_cnt DESC;
```

100 %

Results Messages

	book_id	title	borrow_cnt
1	17	Brief History of Time	2
2	26	Blink	2
3	30	The Subtle Art of Not Giving a F*ck	2

SQLQuery4.sql - DE...65I8\CodeLine (57))*
SQLQuery3.sql - DE...65I8\CodeLine (60))*

WITH per_member_paid AS (
SELECT m.member_id, m.name, SUM(f.paid_amount) AS total_paid
FROM Members m
JOIN Borrowing br ON br.member_id = m.member_id
JOIN Fines f ON f.borrow_id = br.borrow_id
GROUP BY m.member_id, m.name
), avg_paid AS (
SELECT AVG(total_paid*1.0) AS avg_total_paid FROM per_member_paid
)
SELECT p.member_id, p.name, p.total_paid
FROM per_member_paid p
CROSS JOIN avg_paid a
WHERE p.total_paid > a.avg_total_paid
ORDER BY p.total_paid DESC;

100 %

Results Messages

	member_id	name	total_paid
1	12	Faisal Al-Kalbani	5.000
2	6	Khalid Al-Hina	2.000
3	19	Layla Al-Harthy	2.000
4	18	Adil Al-Sinani	1.500

—Thank you—