8. Use set operations to find all authors from California and any publishers that begin with the letter 'N'. Screenshot the result.

```sql
--Use set operations to find all authors from California and any publishers that begin with the letter 'N'. Screenshot the result.
(SELECT authors.au_fname + ' ' + authors.au_lname AS 'Author Name', state FROM authors WHERE state = 'CA')
UNION
(SELECT pub_id, pub_name FROM publishers WHERE pub_name LIKE 'N%');
```

100 % No issues found

T-SQL | Results | Message

	Author Name	state
1	Abraham Bennet	CA
2	Akiko Yokomoto	CA
3	Ann Dull	CA
4	Burt Gringlesby	CA
5	Charlene Locksley	CA
6	Cheryl Carson	CA
7	Dean Straight	CA
8	Dirk Stringer	CA
9	Heather McBadden	CA
10	Johnson White	CA
11	Livia Karsen	CA
12	Marjorie Green	CA
13	Michael O'Leary	CA
14	Sheryl Hunter	CA
15	Stearns MacFeather	CA
16	0736	New Moon Books

9. Use nested queries to find the average price of titles from the New Moon Books publisher. Screenshot the result.

```sql
--Use nested queries to find the average price of titles from the New Moon Books publisher. Screenshot the result.
SELECT avg(price) AS 'Average Price'
FROM titles
WHERE pub_id IN(
SELECT pub_id
FROM publishers
WHERE pub_name = 'New Moon Books');
```

100 % No issues found

T-SQL | Results | Message

	Average Price
1	9.784

10. Use join to find the total number of titles for each author. Screenshot the result.

```sql
--Use join to find the total number of titles for each author. Screenshot the result.
SELECT a.au_id, count(DISTINCT ta.title_id) AS 'Number Of Titles'
FROM authors a JOIN titleauthor ta ON a.au_id = ta.au_id
GROUP BY a.au_id;
```

100 % No issues found

T-SQL | Results | Message

	au_id	Number Of Titles
1	172-32-1176	1
2	213-46-8915	2
3	238-95-7766	1
4	267-41-2394	2
5	274-80-9391	1
6	409-56-7008	1
7	427-17-2319	1
8	472-27-2349	1
9	486-29-1786	1
10	648-92-1872	1
11	672-71-3249	1
12	712-45-1867	1
13	722-51-5454	1
14	724-80-9391	2
15	756-30-7391	1
16	807-91-6654	1
17	846-92-7186	1
18	899-46-2035	2
19	998-72-3567	2

11. Create the relation `discounts`. The data types of the relation attributes are:

`dis_id int`

`discounttype varchar(40)`

`stor_id char(4)`

`title_id varchar(6)`

`discount decimal(4,2)`

The primary key is dis_id ensure that the id value increases by one. The foreign keys are stor_id and title_id ensure cascading for updates and deletions. Verify that discount is a positive value.

Perform a single insert in the new table. Screenshot the inserted data.

```sql
--Create the relation discounts. The data types of the relation attributes are:
--dis_id int
--discounttype varchar(40)
--stor_id char(4)
--title_id varchar(6)
--discount decimal(4,2)

--The primary key is dis_id ensure that the id value increases by one.
--The foreign keys are stor_id and title_id ensure cascading for updates and deletions.
--Verify that discount is a positive value.
--Perform a single insert in the new table. Screenshot the inserted data.
CREATE Table discounts (
dis_id INT IDENTITY(1,1),
discounttype VARCHAR(40),
stor_id char(4),
title_id varchar(6),
discount decimal(4,2),
PRIMARY KEY (dis_id),
CONSTRAINT FK_stor_id
FOREIGN KEY (stor_id) REFERENCES stores(stor_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
CONSTRAINT FK_title_id
FOREIGN KEY (title_id) REFERENCES titles(title_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
CONSTRAINT CHK_discount
    CHECK(discount >= 0)
);
```

100 % No issues found

T-SQL Message

```
Command(s) completed successfully.
```

```sql
--Create the relation discounts. The data types of the relation attributes are:
--dis_id int
--discounttype varchar(40)
--stor_id char(4)
--title_id varchar(6)
--discount decimal(4,2)

--The primary key is dis_id ensure that the id value increases by one.
--The foreign keys are stor_id and title_id ensure cascading for updates and deletions.
--Verify that discount is a positive value.
--Perform a single insert in the new table. Screenshot the inserted data.
CREATE Table discounts (
dis_id INT IDENTITY(1,1),
discounttype VARCHAR(40),
stor_id char(4),
title_id varchar(6),
discount decimal(4,2),
PRIMARY KEY (dis_id),
CONSTRAINT FK_stor_id
FOREIGN KEY (stor_id) REFERENCES stores(stor_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
CONSTRAINT FK_title_id
FOREIGN KEY (title_id) REFERENCES titles(title_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
CONSTRAINT CHK_discount
    CHECK(discount >= 0)
);

INSERT INTO discounts(discounttype, stor_id, title_id, discount) VALUES ('Annual', 6380, 'BU1032', 25.43);
```

100 % No issues found

T-SQL Message

```
(1 row(s) affected)
```

dbo.discounts [Data] | dbo.titles [Data] | dbo.stores [Data] | SQLQuery2.sql *

Max Rows: 1000

dis_id	discounttype	stor_id	title_id	discount
1	Annual	6380	BU1032	25.43
NULL	NULL	NULL	NULL	NULL

12. Create a view named TitleDetails that lists the title, price, and AuthorFullName for titles greater or equal to $10.00. Screenshot all entries in the view displayed from least expensive to most expensive.

```sql
--Create a view named TitleDetails that lists the title, price, and AuthorFullName for titles greater or equal to $10.00.
--Screenshot all entries in the view displayed from least expensive to most expensive.
go
CREATE VIEW [TitleDetails] AS
    SELECT t.title, t.price, a.au_fname + ' ' + a.au_lname AS 'Author Full Name'
    FROM authors a JOIN titleauthor ta ON a.au_id = ta.au_id
    JOIN titles t ON ta.title_id = t.title_id
    WHERE t.price >= 10;
go
```

100 % No issues found

T-SQL Message

```
Command(s) completed successfully.
```

```sql
--Create a view named TitleDetails that lists the title, price, and AuthorFullName for titles greater or equal to $10.00.
--Screenshot all entries in the view displayed from least expensive to most expensive.
go
CREATE VIEW [TitleDetails] AS
    SELECT t.title, t.price, a.au_fname + ' ' + a.au_lname AS 'Author Full Name'
    FROM authors a JOIN titleauthor ta ON a.au_id = ta.au_id
    JOIN titles t ON ta.title_id = t.title_id
    WHERE t.price >= 10;
go

SELECT * from dbo.[TitleDetails];
```

100 % No issues found

T-SQL Results Message

	title	price	Author Full Name
1	Prolonged Data Deprivation: Four Case Studies	19.99	Johnson White
2	The Busy Executive's Database Guide	19.99	Marjorie Green
3	But Is It User Friendly?	22.95	Cheryl Carson
4	Cooking with Computers: Surreptitious Balance Sh...	11.95	Michael O'Leary
5	Sushi, Anyone?	14.99	Michael O'Leary
6	Straight Talk About Computers	19.99	Dean Straight
7	The Busy Executive's Database Guide	19.99	Abraham Bennet
8	Secrets of Silicon Valley	20.00	Ann Dull
9	Sushi, Anyone?	14.99	Burt Gringlesby
10	Fifty Years in Buckingham Palace Kitchens	11.95	Reginald Blotchet-Halls
11	Sushi, Anyone?	14.99	Akiko Yokomoto
12	Silicon Valley Gastronomic Treats	19.99	Innes del Castillo
13	Cooking with Computers: Surreptitious Balance Sh...	11.95	Stearns MacFeather
14	Computer Phobic AND Non-Phobic Individuals: Be...	21.59	Stearns MacFeather
15	Computer Phobic AND Non-Phobic Individuals: Be...	21.59	Livia Karsen
16	Onions, Leeks, and Garlic: Cooking Secrets of the ...	20.95	Sylvia Panteley
17	Secrets of Silicon Valley	20.00	Sheryl Hunter
18	Is Anger the Enemy?	10.95	Anne Ringer
19	Is Anger the Enemy?	10.95	Albert Ringer