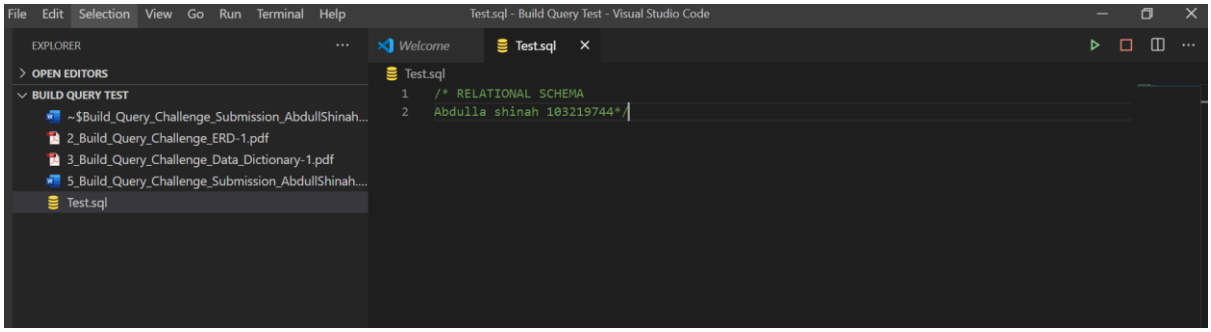
Build Query Challenge Submission Template

Student ID:103219744

Name: Abdulla Shinah

Setup Screenshot



Task 1 Screenshot

Demo.sql

```sql
CREATE DATABASE demo;
USE demo;

/* RELATIONAL SCHEMA
Abdulla shinah 103219744

Task



Tour (TourName, Description)
PRIMARY KEY (TourName)

Client (ClientID, Surname, GivenName, Gender)
PRIMARY KEY (ClientID)

Event (TourName, EventYear, EventMonth, EventDay, Fee)
PRIMARY KEY (TourName, EventYear, EventMonth, EventDay)
FOREIGN KEY (TourName) references Tour

Booking (ClientID, TourName, EventYear, EventMonth, EventDay DateBooked, Payment)
PRIMARY KEY (ClientID, TourName, EventYear, EventMonth, EventDay)
FOREIGN KEY (TourName, EventYear, EventMonth, EventDay) References Event
FOREIGN KEY (ClientID) References Client
```

Task 2 Screenshot

Table Client

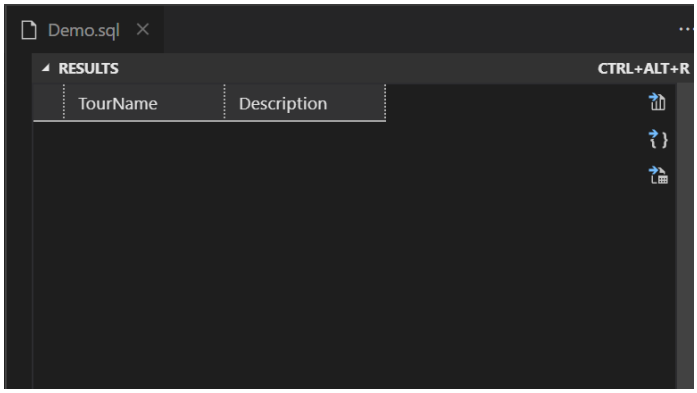
RESULTS

	database	schema	name	type
1	demo	dbo	TASK5	VIEW
2	demo	dbo	Tour	BASE TABLE
3	demo	dbo	Client	BASE TABLE
4	demo	dbo	Event	BASE TABLE
5	demo	dbo	Booking	BASE TABLE

```
SELECT *
FROM INFORMATION_SCHEMA.tables;
```

```
SELECT table_catalog [database], table_schema [schema], table_name name, table
_type type
FROM INFORMATION_SCHEMA.TABLES
GO
```

Table Tour



Build Query Challenge Submission Template

Table Event

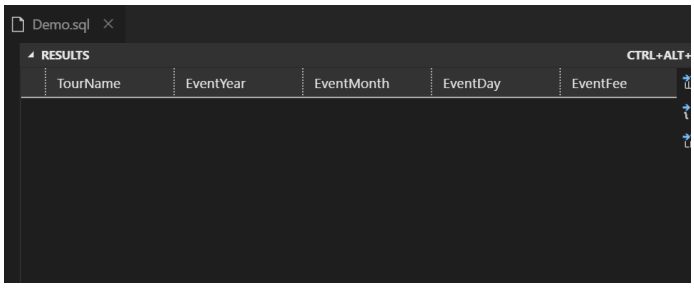
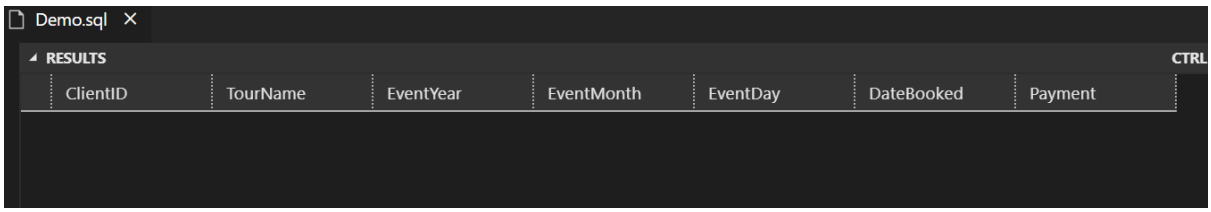
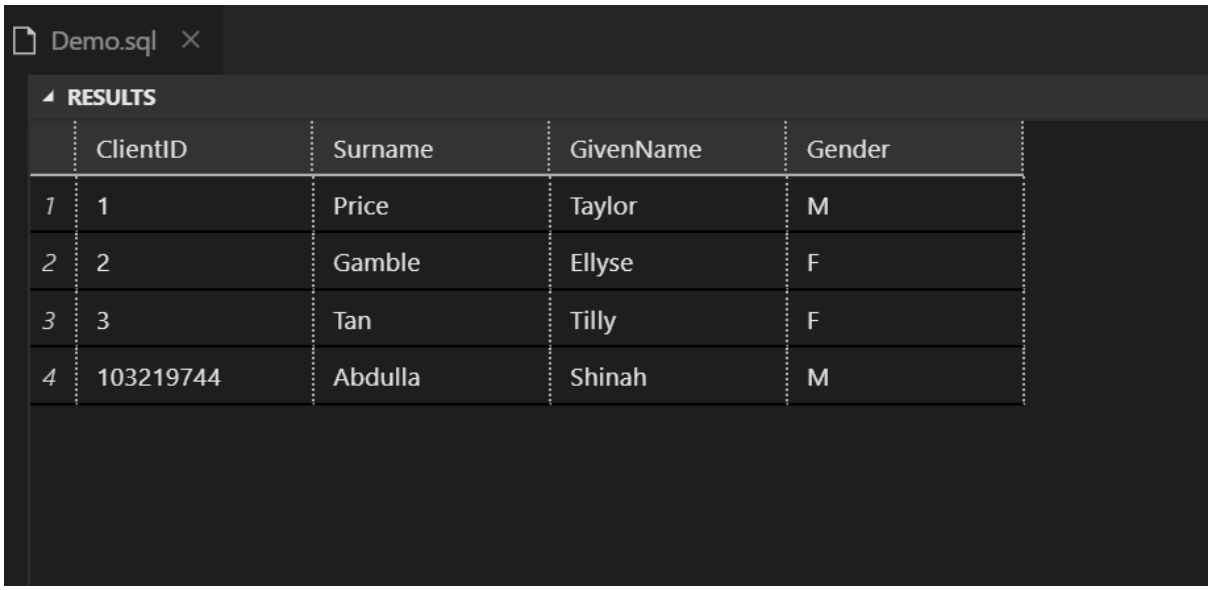


Table Booking



Task 3 Screenshot

```
Select * from client
```



Task 4 Screenshot

Query 1 (since query result was big took 2 screen shots)

RESULTS

	GivenName	Surname	TourName	Description	EventDay
1	Taylor	Price	North	Tour of winerie...	13
2	Taylor	Price	North	Tour of winerie...	9
3	Taylor	Price	South	Tour of winerie...	9
4	Ellyse	Gamble	North	Tour of winerie...	13
5	Ellyse	Gamble	North	Tour of winerie...	9
6	Ellyse	Gamble	South	Tour of winerie...	16
7	Ellyse	Gamble	West	Tour of winerie...	29
8	Tilly	Tan	North	Tour of winerie...	13
9	Tilly	Tan	South	Tour of winerie...	16
10	Tilly	Tan	West	Tour of winerie...	29
11	Shinah	Abdulla	North	Tour of winerie...	9
12	Shinah	Abdulla	South	Tour of winerie...	16
13	Shinah	Abdulla	West	Tour of winerie...	29

EventYear	EventMonth	EventFee	Payment	DateBooked
2016	Feb	225.0000	225.0000	2016-01-08
2016	Jan	200.0000	200.0000	2015-12-10
2016	Jan	200.0000	200.0000	2015-12-10
2016	Feb	225.0000	125.0000	2016-01-14
2016	Jan	200.0000	200.0000	2015-12-16
2016	Jan	200.0000	200.0000	2015-12-18
2016	Jan	225.0000	225.0000	2015-12-17
2016	Feb	225.0000	225.0000	2016-02-03
2016	Jan	200.0000	200.0000	2016-01-09
2016	Jan	225.0000	200.0000	2015-12-18
2016	Jan	200.0000	200.0000	2015-12-16
2016	Jan	200.0000	200.0000	2015-12-16
2016	Jan	225.0000	255.0000	2015-12-16

```sql
--Task 4 Query 1 Write a query that shows the client first name and surname,
--the tour name and description,the tour event year, month, day and fee, the booking date and the fee paid for

SELECT Client.GivenName , CLIENT.Surname , TOUR.TourName , TOUR.Description , EVENT.EventDay , EVENT.EventYear
FROM TOUR
INNER JOIN EVENT
ON TOUR.TourName = EVENT.TourName
INNER JOIN BOOKING
ON (EVENT.EventYear = BOOKING.EventYear AND EVENT.EventMonth = BOOKING.EventMonth AND  EVENT.EventDay = BOOKIN
INNER JOIN CLIENT
ON BOOKING.ClientID = CLIENT.ClientID;
```

Build Query Challenge Submission Template

Query 2

RESULTS

	EventMonth	TourName	Num Booking
1	Jan	North	3
2	Jan	South	4
3	Jan	West	3
4	Feb	North	3

```
--Task 4 Query 2 --- Write a query which shows the number of bookings for each (tour event) month, for each
--tour in the following example format


SELECT BOOKING.EventMonth, BOOKING.TourName, COUNT(*) AS [Num Booking]
FROM BOOKING
GROUP BY BOOKING.EventMonth, BOOKING.TourName
ORDER BY BOOKING.EventMonth DESC, BOOKING.TourName
```

Query 3

RESULTS CTRL

	ClientID	TourName	EventYear	EventMonth	EventDay	DateBooked	Payment
1	1	North	2016	Feb	13	2016-01-08	225.0000
2	2	West	2016	Jan	29	2015-12-17	225.0000
3	3	North	2016	Feb	13	2016-02-03	225.0000
4	103219744	West	2016	Jan	29	2015-12-16	255.0000

```
--Task 4 Query 3:Write a query which lists all bookings which have a payment
--amount greater than the average payment amount. (This query must use a sub-query.)

SELECT *
FROM BOOKING
WHERE Payment > ( SELECT AVG( Payment ) FROM BOOKING );
```

Task 5 Screenshot

```
--Task 5 Create a View based on Query 1 from Task 4--

CREATE VIEW TASK5 ASSELECT C.GivenName, C.Surname, T.TourName, T.DESCRIPTION, E.EventYear,
E.EventMonth,E.EventDay,E.EventFee, B.DateBooked, B.Payment
FROM Booking B

INNER JOIN Client C
ON B.ClientID = C.ClientID

INNER JOIN Event E
ON B.TourName = E.TourName AND B.EventYear = E.EventYear AND B.EventMonth = E.EventMonth
AND B.EventDay = E.EventDay

INNER JOIN Tour T
ON E.TourName = T.TourName;
```

Task 6 Screenshot

Test 1

```
-- Return same 13 rows of data as per the orginal query task 4 - Query 1
SELECT *
FROM Booking;
```

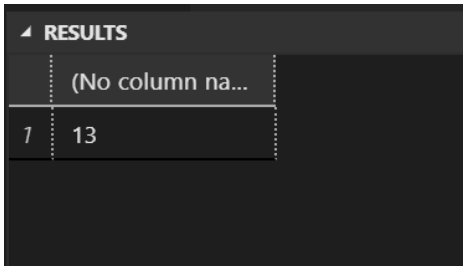
RESULTS CTRL+

	ClientID	TourName	EventYear	EventMonth	EventDay	DateBooked	Payment
1	1	North	2016	Feb	13	2016-01-08	225.0000
2	1	North	2016	Jan	9	2015-12-10	200.0000
3	1	South	2016	Jan	9	2015-12-10	200.0000
4	2	North	2016	Feb	13	2016-01-14	125.0000
5	2	North	2016	Jan	9	2015-12-16	200.0000
6	2	South	2016	Jan	16	2015-12-18	200.0000
7	2	West	2016	Jan	29	2015-12-17	225.0000
8	3	North	2016	Feb	13	2016-02-03	225.0000
9	3	South	2016	Jan	16	2016-01-09	200.0000
10	3	West	2016	Jan	29	2015-12-18	200.0000
11	103219744	North	2016	Jan	9	2015-12-16	200.0000
12	103219744	South	2016	Jan	16	2015-12-16	200.0000
13	103219744	West	2016	Jan	29	2015-12-16	255.0000

Build Query Challenge Submission Template

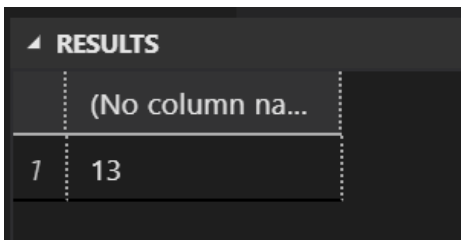
Test 2

```
-- Returns Count of rows as 13 - same number of rows in the task 4 - Query 1
SELECT COUNT(*)
FROM Booking;

-- Can confirm the query result is correct which is showing 13 rows of data----
```



Test 3

```
--Testing Query 2 From Task 4--

-- Both these test queries provide 13 results (Rows of data) ---
SELECT *
FROM Booking;

SELECT COUNT(*)
FROM Booking;

--- The result from the query is ---

Feb North   3
Jan North   3
Jan South   4
Jan West    3

Total = 3+3+4+3 = 13

-- The output of the queries is 13 or 13 rows of data--
```



Build Query Challenge Submission Template

RESULTS CTRL+A

	ClientID	TourName	EventYear	EventMonth	EventDay	DateBooked	Payment
1	1	North	2016	Feb	13	2016-01-08	225.0000
2	1	North	2016	Jan	9	2015-12-10	200.0000
3	1	South	2016	Jan	9	2015-12-10	200.0000
4	2	North	2016	Feb	13	2016-01-14	125.0000
5	2	North	2016	Jan	9	2015-12-16	200.0000
6	2	South	2016	Jan	16	2015-12-18	200.0000
7	2	West	2016	Jan	29	2015-12-17	225.0000
8	3	North	2016	Feb	13	2016-02-03	225.0000
9	3	South	2016	Jan	16	2016-01-09	200.0000
10	3	West	2016	Jan	29	2015-12-18	200.0000
11	103219744	North	2016	Jan	9	2015-12-16	200.0000
12	103219744	South	2016	Jan	16	2015-12-16	200.0000
13	103219744	West	2016	Jan	29	2015-12-16	255.0000

Test 4

```
--Test Query 3 From Task 4--

--The test queries returns count of 4 Rows, which is as the original query Task 4 Query 3 --
SELECT COUNT(Payment)
FROM Booking
WHERE Payment > (SELECT AVG(Payment) FROM Booking);
```

RESULTS

	(No column na...
1	4

```
--calculated the average as 200 from all and the result is the same as Task 4 Query No 3--
SELECT *
FROM Booking
WHERE Payment > 200;
```

Build Query Challenge Submission Template

RESULTS CTRL+ALT+

	ClientID	TourName	EventYear	EventMonth	EventDay	DateBooked	Payment
1	1	North	2016	Feb	13	2016-01-08	225.0000
2	2	West	2016	Jan	29	2015-12-17	225.0000
3	3	North	2016	Feb	13	2016-02-03	225.0000
4	103219744	West	2016	Jan	29	2015-12-16	255.0000