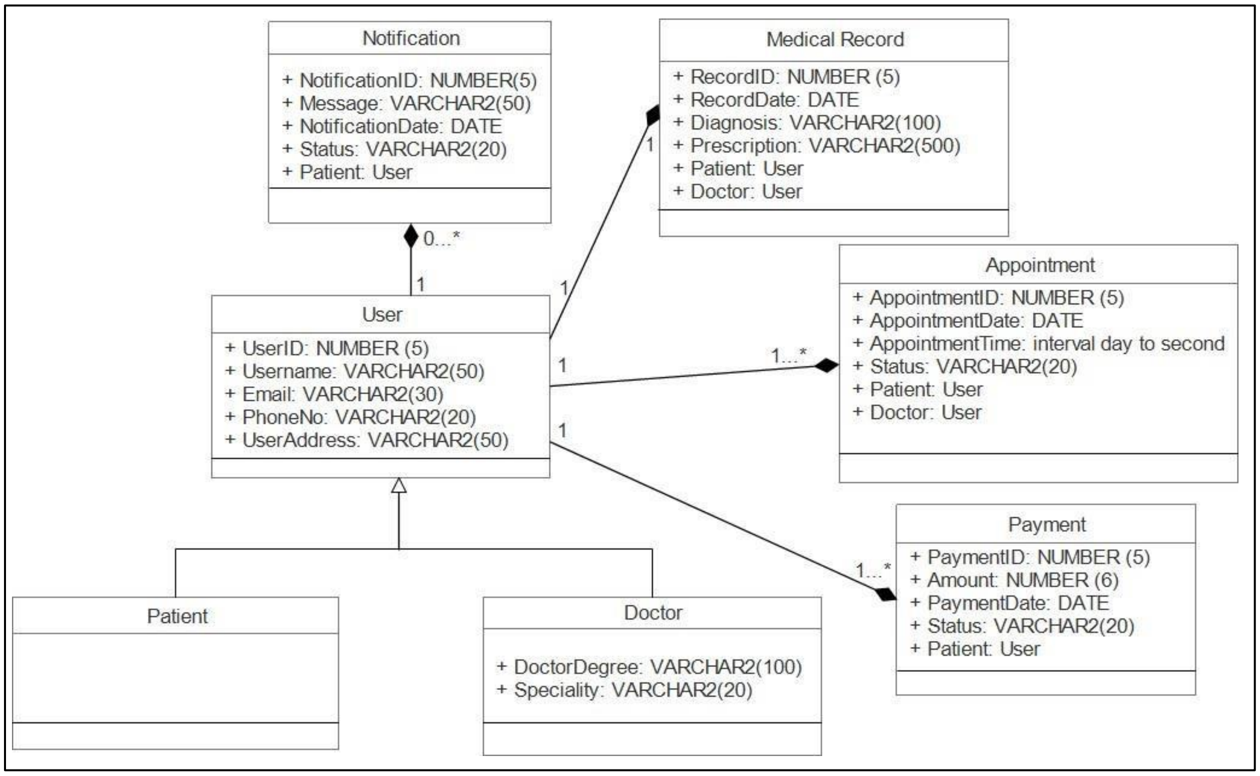
Telemedicine Online Platform Database Design

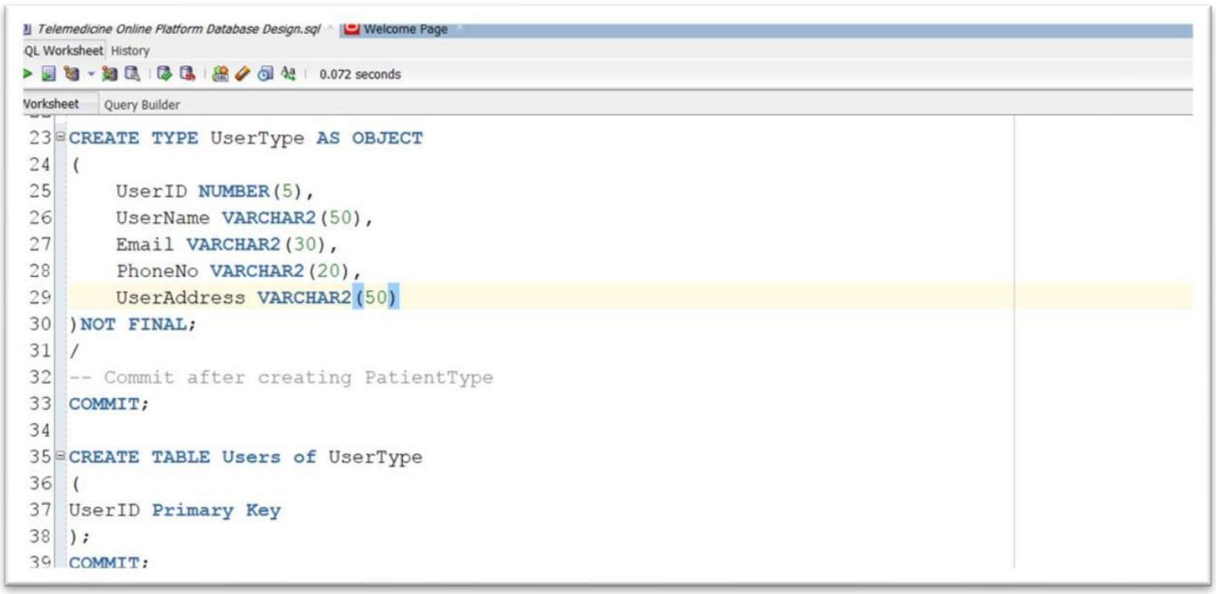
Task one:



Task two (Oracle SQL code screenshots and output):

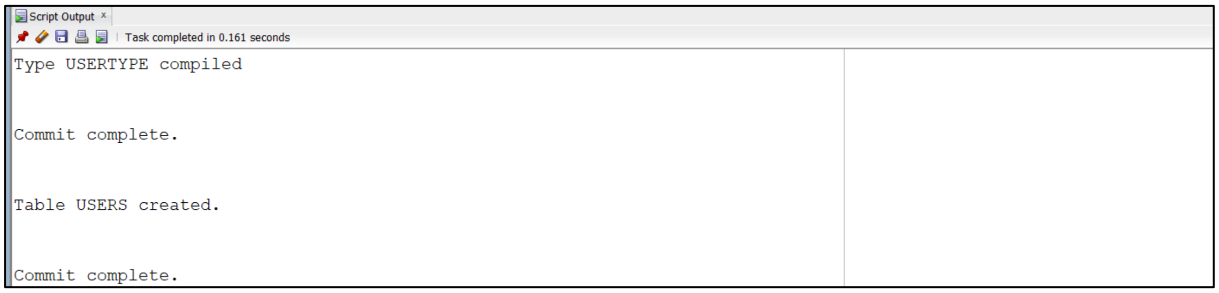
1. User Table

Code of User Table:



```
CREATE TYPE UserType AS OBJECT
(
    UserID NUMBER(5),
    UserName VARCHAR2(50),
    Email VARCHAR2(30),
    PhoneNo VARCHAR2(20),
    UserAddress VARCHAR2(50)
)NOT FINAL;
/
-- Commit after creating PatientType
COMMIT;

CREATE TABLE Users of UserType
(
UserID Primary Key
);
COMMIT;
```

Script Output:

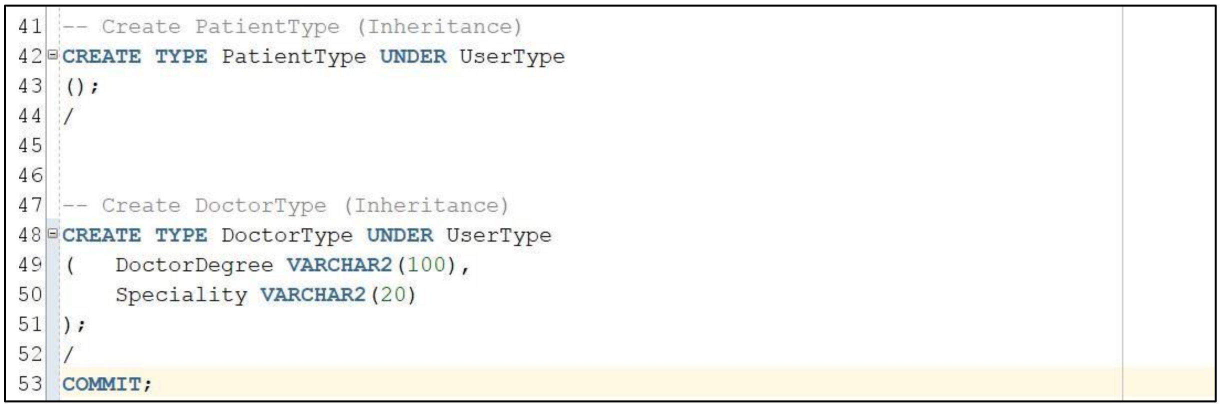


```
Type USERTYPE compiled


Commit complete.


Table USERS created.


Commit complete.
```

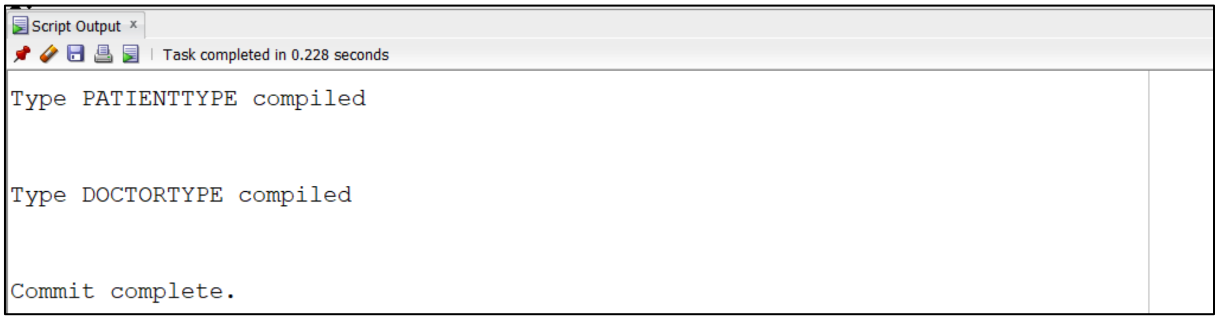
Code of Inheritances of User Table:

Inheritance: PatientType

Inheritance: UserType

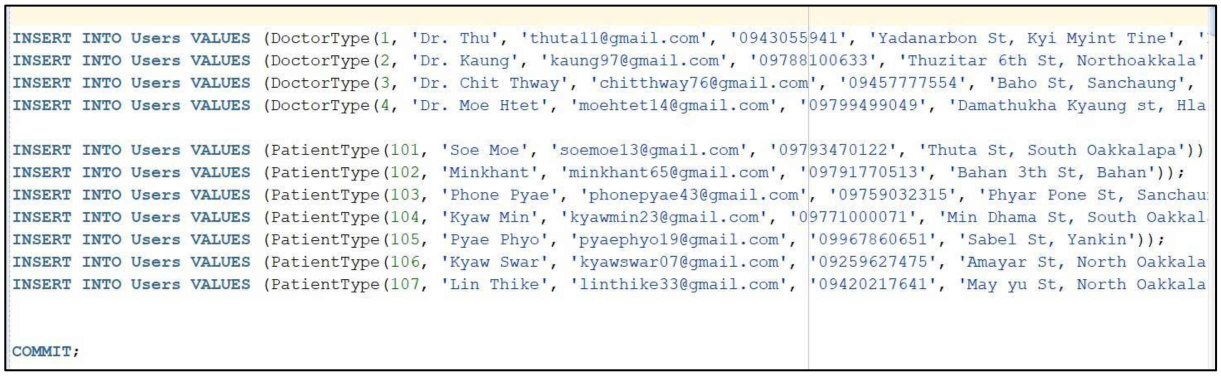

```
-- Create PatientType (Inheritance)
CREATE TYPE PatientType UNDER UserType
();
/


-- Create DoctorType (Inheritance)
CREATE TYPE DoctorType UNDER UserType
(   DoctorDegree VARCHAR2(100),
    Speciality VARCHAR2(20)
);
/
COMMIT;
```

Script Output:

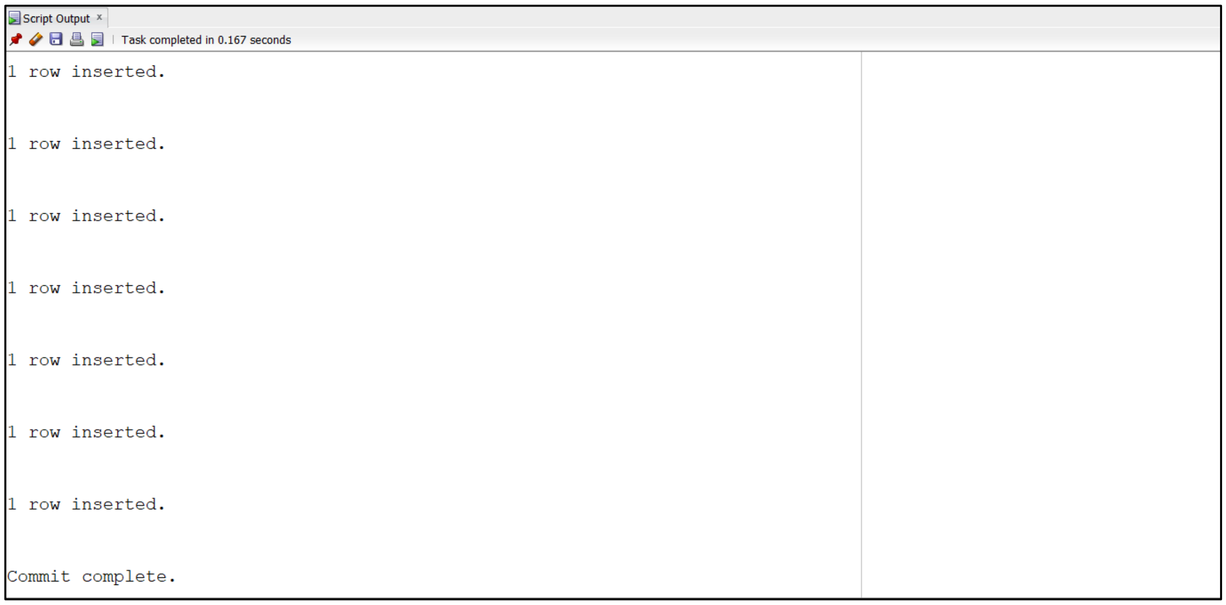


```
Type PATIENTTYPE compiled


Type DOCTORTYPE compiled


Commit complete.
```

Code of Insert Data of User Table:

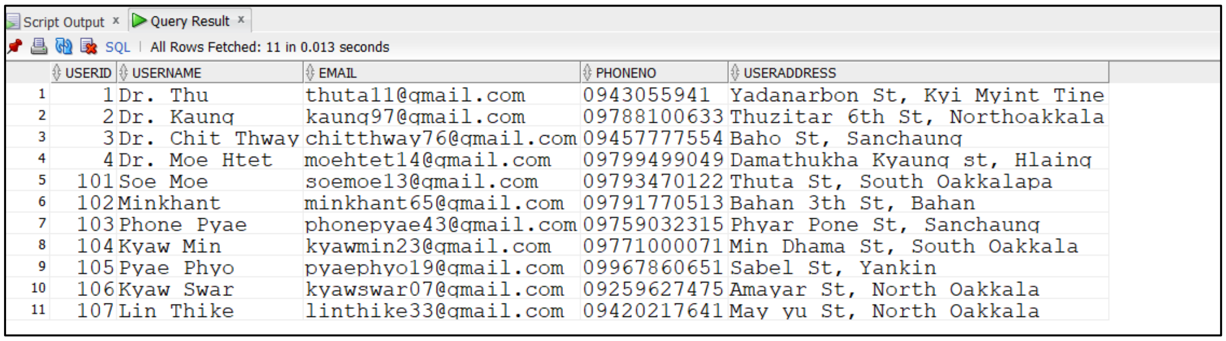

```
INSERT INTO Users VALUES (DoctorType(1, 'Dr. Thu', 'thuta11@gmail.com', '0943055941', 'Yadanarbon St, Kyi Myint Tine', '
INSERT INTO Users VALUES (DoctorType(2, 'Dr. Kaung', 'kaung97@gmail.com', '09788100633', 'Thuzitar 6th St, Northoakkala'
INSERT INTO Users VALUES (DoctorType(3, 'Dr. Chit Thway', 'chitthway76@gmail.com', '09457777554', 'Baho St, Sanchaung',
INSERT INTO Users VALUES (DoctorType(4, 'Dr. Moe Htet', 'moehtet14@gmail.com', '09799499049', 'Damathukha Kyaung st, Hla

INSERT INTO Users VALUES (PatientType(101, 'Soe Moe', 'soemoe13@gmail.com', '09793470122', 'Thuta St, South Oakkalapa'))
INSERT INTO Users VALUES (PatientType(102, 'Minkhant', 'minkhant65@gmail.com', '09791770513', 'Bahan 3th St, Bahan'));
INSERT INTO Users VALUES (PatientType(103, 'Phone Pyae', 'phonepyae43@gmail.com', '09759032315', 'Phyar Pone St, Sanchau
INSERT INTO Users VALUES (PatientType(104, 'Kyaw Min', 'kyawmin23@gmail.com', '09771000071', 'Min Dhama St, South Oakkal
INSERT INTO Users VALUES (PatientType(105, 'Pyae Phyo', 'pyaephyo19@gmail.com', '09967860651', 'Sabel St, Yankin'));
INSERT INTO Users VALUES (PatientType(106, 'Kyaw Swar', 'kyawswar07@gmail.com', '09259627475', 'Amayar St, North Oakkala
INSERT INTO Users VALUES (PatientType(107, 'Lin Thike', 'linthike33@gmail.com', '09420217641', 'May yu St, North Oakkala


COMMIT;
```

Script Output of Insert Data of User Table:



```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


Commit complete.
```

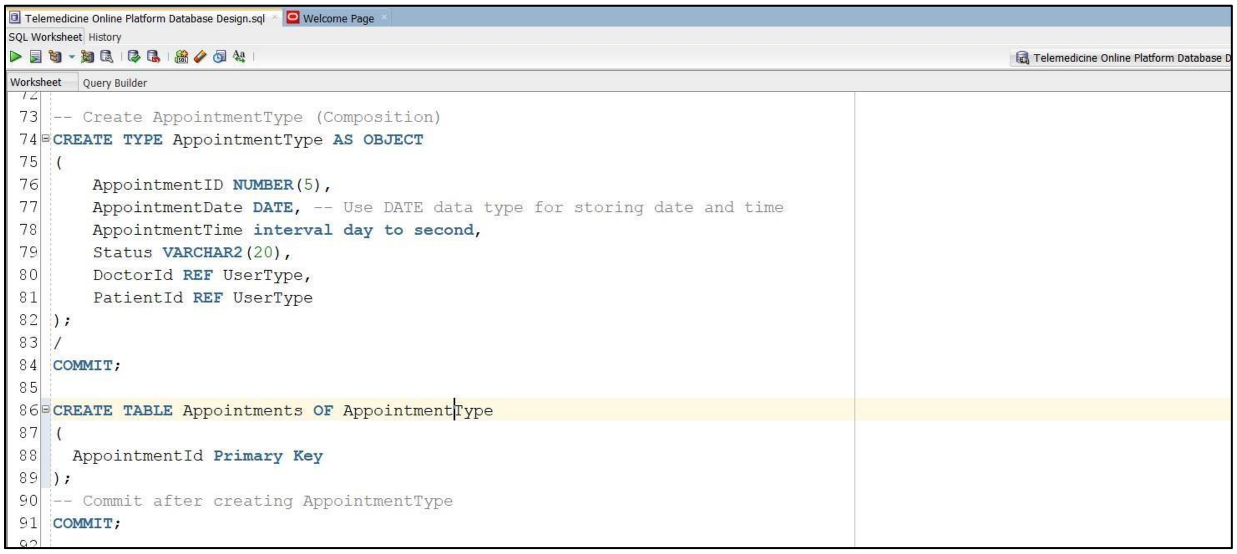
Query Result of User Table:



	USERID	USERNAME	EMAIL	PHONENO	USERADDRESS
1	1	Dr. Thu	thuta11@gmail.com	0943055941	Yadanarbon St, Kyi Myint Tine
2	2	Dr. Kaung	kaung97@gmail.com	09788100633	Thuzitar 6th St, Northoakkala
3	3	Dr. Chit Thway	chitthway76@gmail.com	09457777554	Baho St, Sanchaung
4	4	Dr. Moe Htet	moehtet14@gmail.com	09799499049	Damathukha Kyaung st, Hlaing
5	101	Soe Moe	soemoe13@gmail.com	09793470122	Thuta St, South Oakkalapa
6	102	Minkhant	minkhant65@gmail.com	09791770513	Bahan 3th St, Bahan
7	103	Phone Pyae	phonepyae43@gmail.com	09759032315	Phyar Pone St, Sanchaung
8	104	Kyaw Min	kyawmin23@gmail.com	09771000071	Min Dhama St, South Oakkala
9	105	Pyae Phyo	pyaephyo19@gmail.com	09967860651	Sabel St, Yankin
10	106	Kyaw Swar	kyawswar07@gmail.com	09259627475	Amayar St, North Oakkala
11	107	Lin Thike	linthike33@gmail.com	09420217641	May yu St, North Oakkala

2. Appointments Table

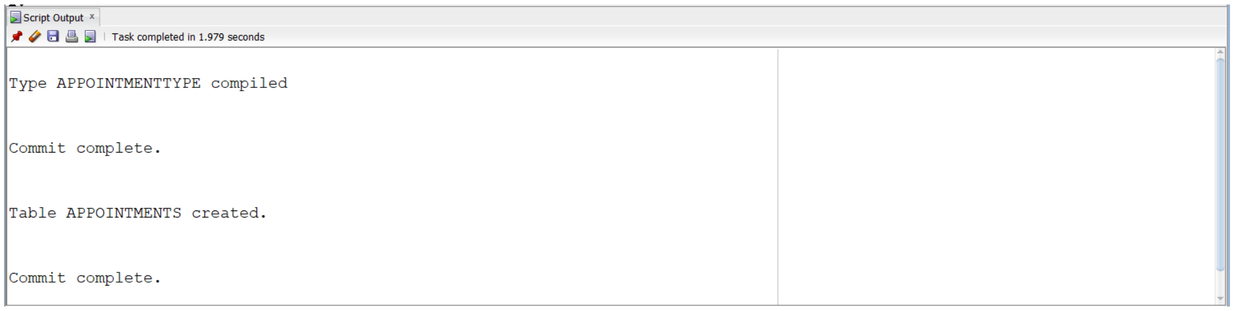
Code of Appointments Table:



Telemedicine Online Platform Database Design.sql
Welcome Page
SQL Worksheet
History
Telemedicine Online Platform Database D
Worksheet
Query Builder

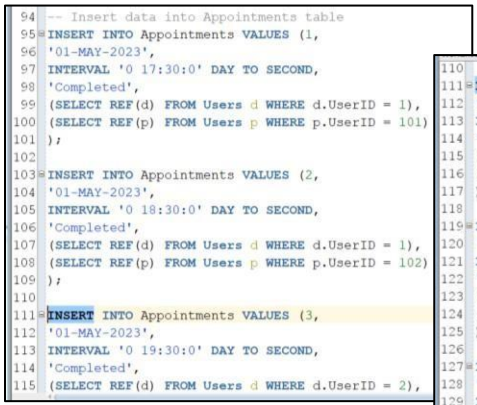
```
-- Create AppointmentType (Composition)
CREATE TYPE AppointmentType AS OBJECT
(
    AppointmentID NUMBER(5),
    AppointmentDate DATE, -- Use DATE data type for storing date and time
    AppointmentTime interval day to second,
    Status VARCHAR2(20),
    DoctorId REF UserType,
    PatientId REF UserType
);
/
COMMIT;

CREATE TABLE Appointments OF AppointmentType
(
  AppointmentId Primary Key
);
-- Commit after creating AppointmentType
COMMIT;
```

Script Output:

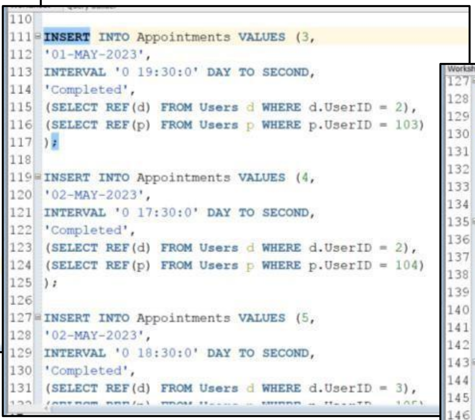


```
Type APPOINTMENTTYPE compiled


Commit complete.


Table APPOINTMENTS created.


Commit complete.
```

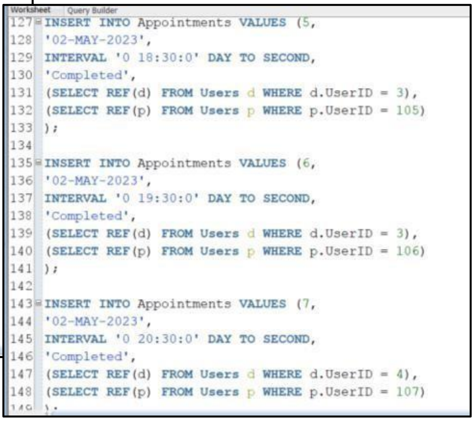
Code of Insert Data of Appointments Table:



```
-- Insert data into Appointments table
INSERT INTO Appointments VALUES (1,
'01-MAY-2023',
INTERVAL '0 17:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 1),
(SELECT REF(p) FROM Users p WHERE p.UserID = 101)
);

INSERT INTO Appointments VALUES (2,
'01-MAY-2023',
INTERVAL '0 18:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 1),
(SELECT REF(p) FROM Users p WHERE p.UserID = 102)
);

INSERT INTO Appointments VALUES (3,
'01-MAY-2023',
INTERVAL '0 19:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 2),
```

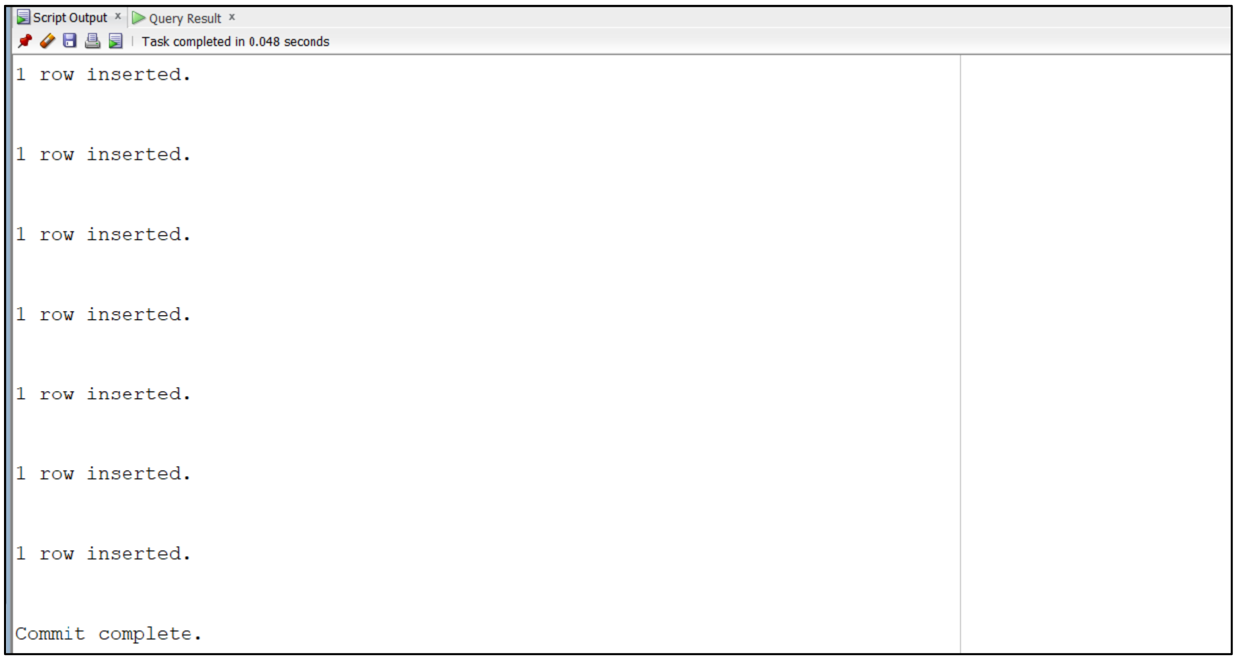


```

INSERT INTO Appointments VALUES (3,
'01-MAY-2023',
INTERVAL '0 19:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 2),
(SELECT REF(p) FROM Users p WHERE p.UserID = 103)
);

INSERT INTO Appointments VALUES (4,
'02-MAY-2023',
INTERVAL '0 17:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 2),
(SELECT REF(p) FROM Users p WHERE p.UserID = 104)
);

INSERT INTO Appointments VALUES (5,
'02-MAY-2023',
INTERVAL '0 18:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 3),
```



```
INSERT INTO Appointments VALUES (5,
'02-MAY-2023',
INTERVAL '0 18:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 3),
(SELECT REF(p) FROM Users p WHERE p.UserID = 105)
);

INSERT INTO Appointments VALUES (6,
'02-MAY-2023',
INTERVAL '0 19:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 3),
(SELECT REF(p) FROM Users p WHERE p.UserID = 106)
);

INSERT INTO Appointments VALUES (7,
'02-MAY-2023',
INTERVAL '0 20:30:0' DAY TO SECOND,
'Completed',
(SELECT REF(d) FROM Users d WHERE d.UserID = 4),
(SELECT REF(p) FROM Users p WHERE p.UserID = 107)
```

Script Output of Insert Data of Appointments Table:

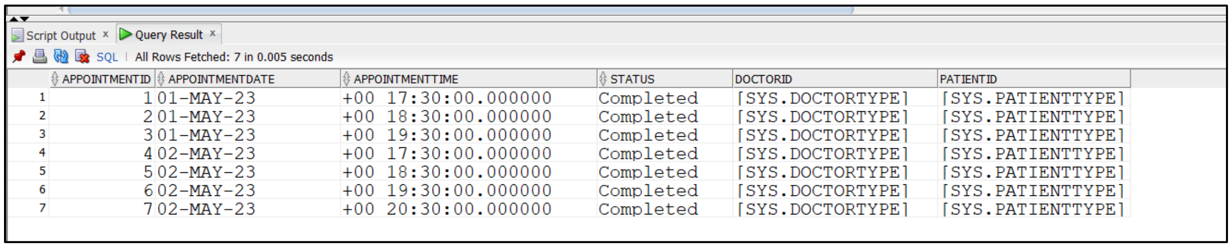


Script Output × Query Result ×

Task completed in 0.048 seconds

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


Commit complete.
```

Query Result of Appointments Table:



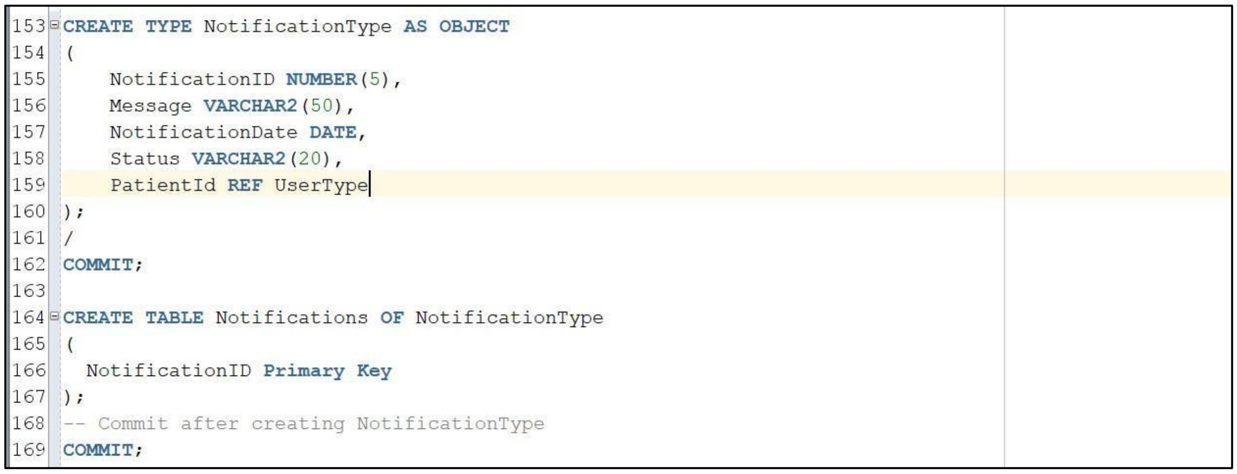
Script Output × Query Result ×

SQL | All Rows Fetched: 7 in 0.005 seconds

	APPOINTMENTID	APPOINTMENTDATE	APPOINTMENTTIME	STATUS	DOCTORID	PATIENTID
1	1	01-MAY-23	+00 17:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]
2	2	01-MAY-23	+00 18:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]
3	3	01-MAY-23	+00 19:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]
4	4	02-MAY-23	+00 17:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]
5	5	02-MAY-23	+00 18:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]
6	6	02-MAY-23	+00 19:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]
7	7	02-MAY-23	+00 20:30:00.000000	Completed	[SYS.DOCTORTYPE]	[SYS.PATIENTTYPE]

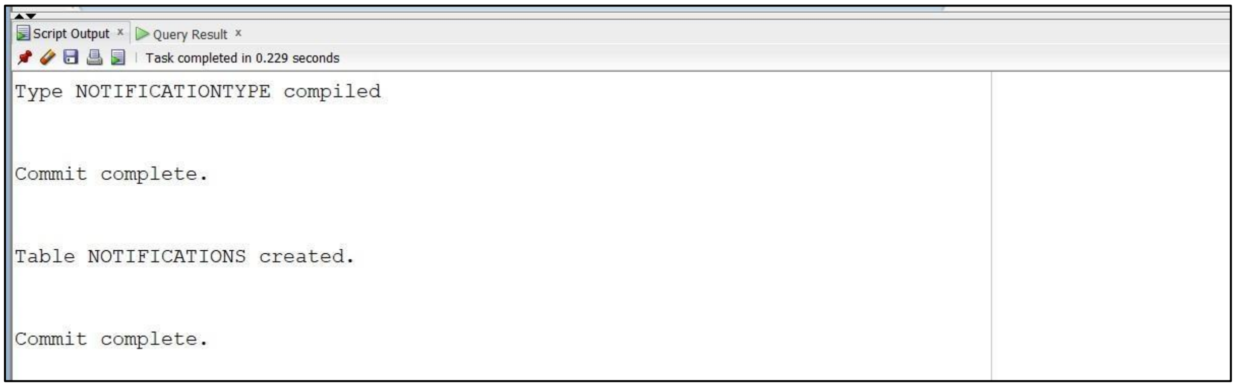
3. Notifications Table

Code of Notifications Table:



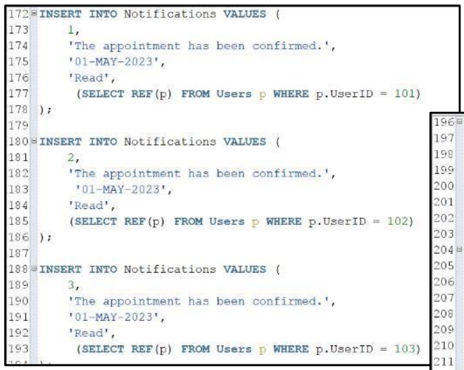
```
CREATE TYPE NotificationType AS OBJECT
(
    NotificationID NUMBER(5),
    Message VARCHAR2(50),
    NotificationDate DATE,
    Status VARCHAR2(20),
    PatientId REF UserType
);
/
COMMIT;

CREATE TABLE Notifications OF NotificationType
(
  NotificationID Primary Key
);
-- Commit after creating NotificationType
COMMIT;
```

Script Output:

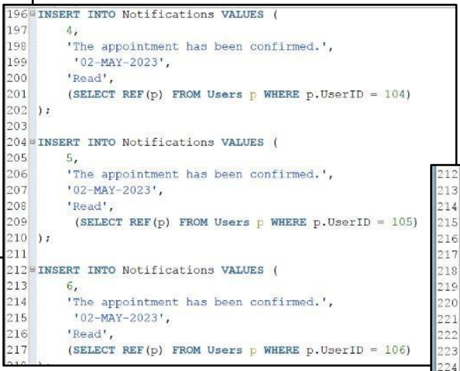


```
Type NOTIFICATIONTYPE compiled


Commit complete.


Table NOTIFICATIONS created.


Commit complete.
```

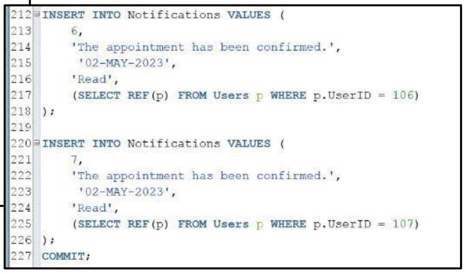
Code of Insert Data of Notifications Table:



```
INSERT INTO Notifications VALUES (
    1,
    'The appointment has been confirmed.',
    '01-MAY-2023',
    'Read',
     (SELECT REF(p) FROM Users p WHERE p.UserID = 101)
);

INSERT INTO Notifications VALUES (
    2,
    'The appointment has been confirmed.',
     '01-MAY-2023',
    'Read',
    (SELECT REF(p) FROM Users p WHERE p.UserID = 102)
);

INSERT INTO Notifications VALUES (
    3,
    'The appointment has been confirmed.',
    '01-MAY-2023',
    'Read',
     (SELECT REF(p) FROM Users p WHERE p.UserID = 103)
```

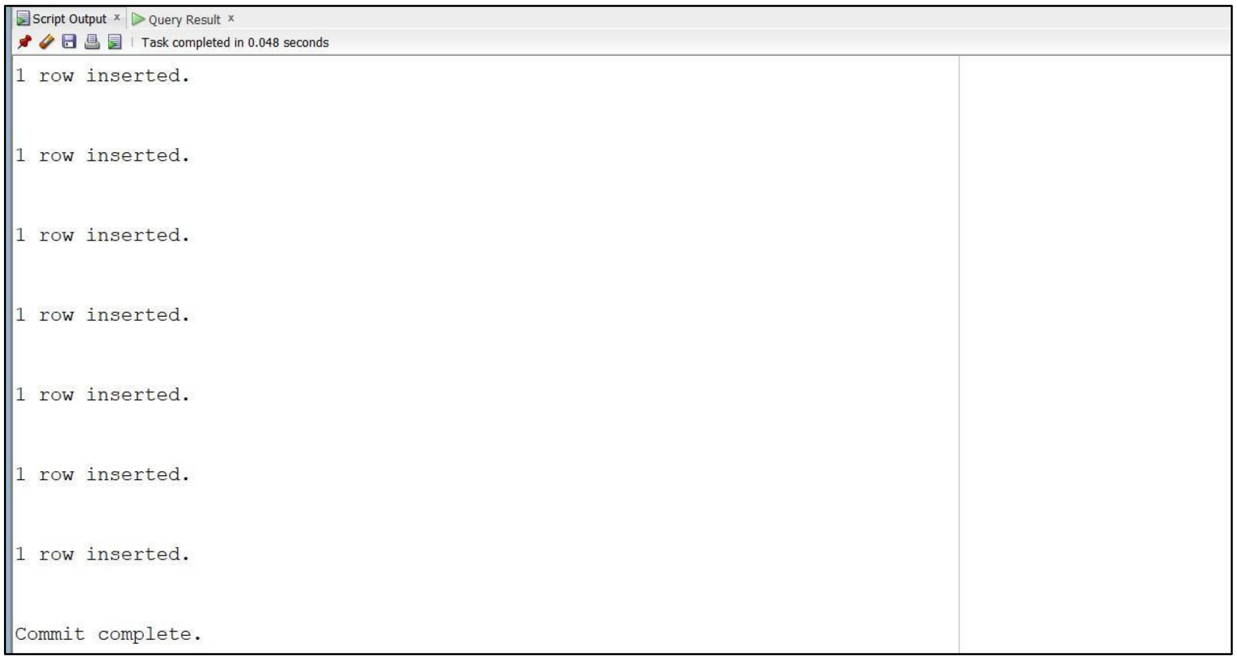


```
INSERT INTO Notifications VALUES (
    4,
    'The appointment has been confirmed.',
     '02-MAY-2023',
    'Read',
    (SELECT REF(p) FROM Users p WHERE p.UserID = 104)
);

INSERT INTO Notifications VALUES (
    5,
    'The appointment has been confirmed.',
    '02-MAY-2023',
    'Read',
     (SELECT REF(p) FROM Users p WHERE p.UserID = 105)
);

INSERT INTO Notifications VALUES (
    6,
    'The appointment has been confirmed.',
     '02-MAY-2023',
    'Read',
    (SELECT REF(p) FROM Users p WHERE p.UserID = 106)
```



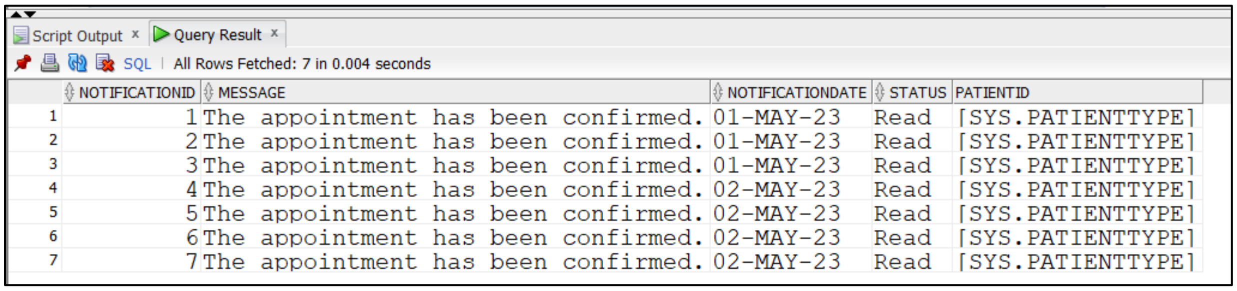
```
INSERT INTO Notifications VALUES (
    6,
    'The appointment has been confirmed.',
     '02-MAY-2023',
    'Read',
    (SELECT REF(p) FROM Users p WHERE p.UserID = 106)
);

INSERT INTO Notifications VALUES (
    7,
    'The appointment has been confirmed.',
     '02-MAY-2023',
    'Read',
    (SELECT REF(p) FROM Users p WHERE p.UserID = 107)
);
COMMIT;
```

Script Output of Insert Data of Notifications Table:



```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


Commit complete.
```

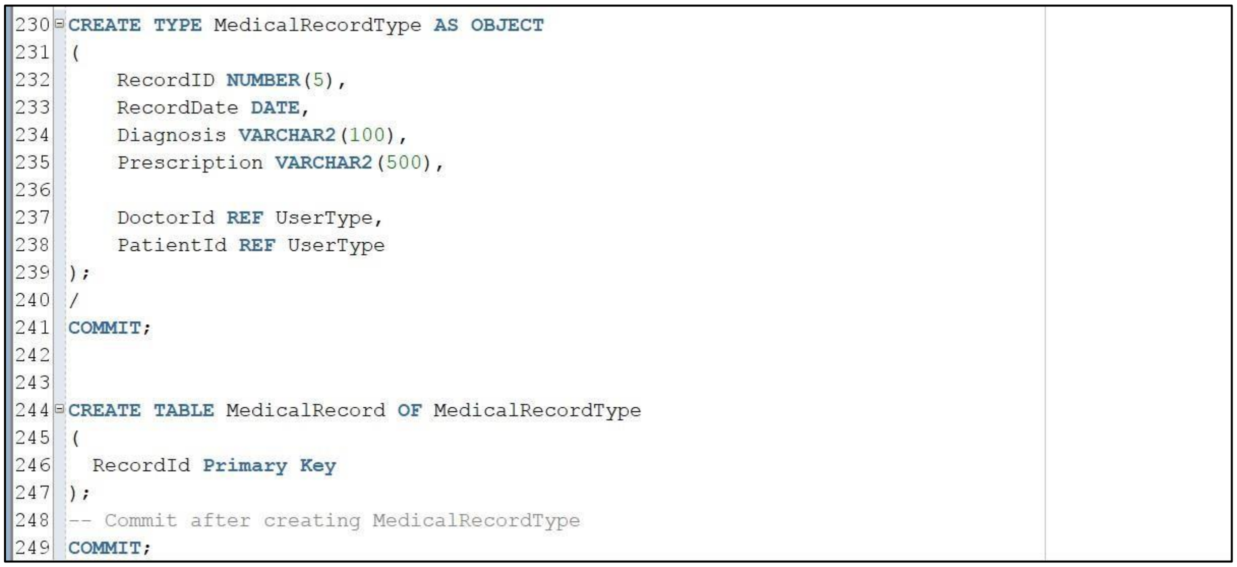
Query Result of Notifications Table:



	NOTIFICATIONID	MESSAGE	NOTIFICATIONDATE	STATUS	PATIENTID
1	1	The appointment has been confirmed.	01-MAY-23	Read	[SYS.PATIENTTYPE]
2	2	The appointment has been confirmed.	01-MAY-23	Read	[SYS.PATIENTTYPE]
3	3	The appointment has been confirmed.	01-MAY-23	Read	[SYS.PATIENTTYPE]
4	4	The appointment has been confirmed.	02-MAY-23	Read	[SYS.PATIENTTYPE]
5	5	The appointment has been confirmed.	02-MAY-23	Read	[SYS.PATIENTTYPE]
6	6	The appointment has been confirmed.	02-MAY-23	Read	[SYS.PATIENTTYPE]
7	7	The appointment has been confirmed.	02-MAY-23	Read	[SYS.PATIENTTYPE]

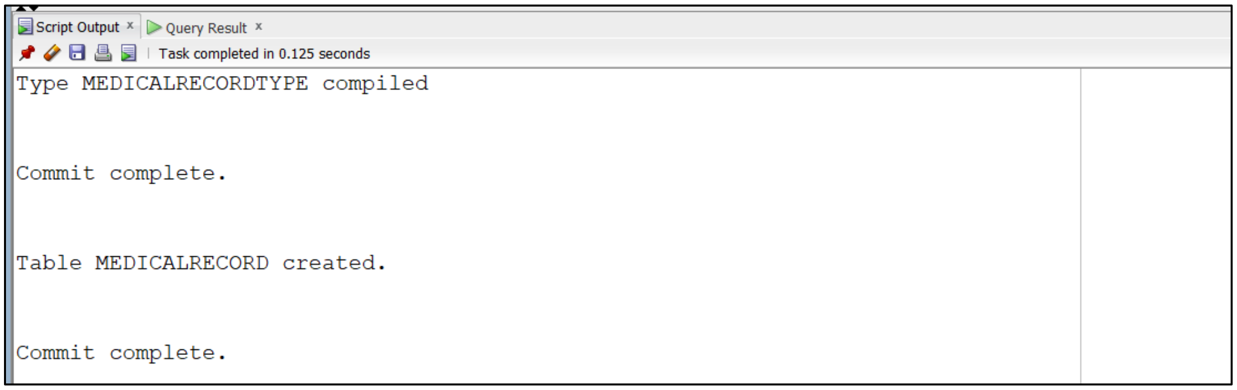
4. MedicalRecord Table

Code of MedicalRecord Table:



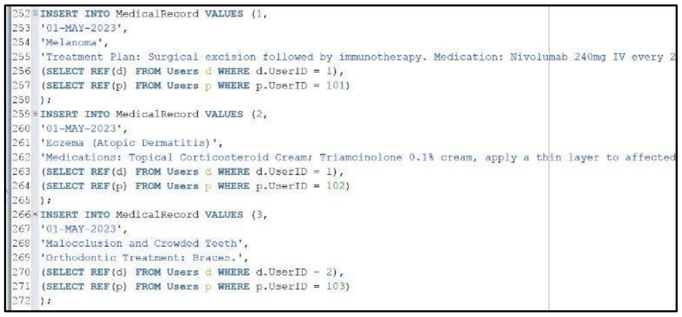
```
CREATE TYPE MedicalRecordType AS OBJECT
(
    RecordID NUMBER(5),
    RecordDate DATE,
    Diagnosis VARCHAR2(100),
    Prescription VARCHAR2(500),

    DoctorId REF UserType,
    PatientId REF UserType
);
/
COMMIT;


CREATE TABLE MedicalRecord OF MedicalRecordType
(
  RecordId Primary Key
);
-- Commit after creating MedicalRecordType
COMMIT;
```

Script Output:

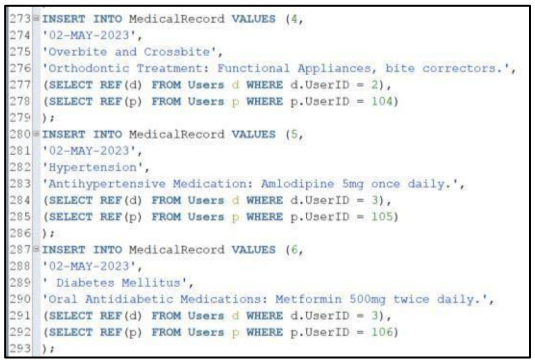


```
Type MEDICALRECORDTYPE compiled


Commit complete.


Table MEDICALRECORD created.


Commit complete.
```

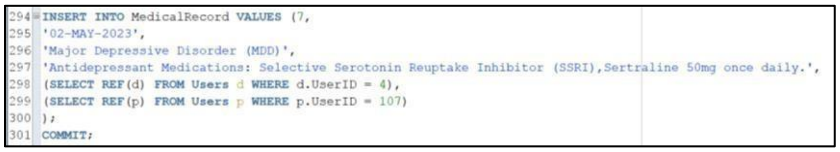
Code of Insert Data of MedicalRecord Table:



```
INSERT INTO MedicalRecord VALUES (1,
'01-MAY-2023',
'Melanoma',
'Treatment Plan: Surgical excision followed by immunotherapy. Medication: Nivolumab 240mg IV every 2
(SELECT REF(d) FROM Users d WHERE d.UserID = 1),
(SELECT REF(p) FROM Users p WHERE p.UserID = 101)
);
INSERT INTO MedicalRecord VALUES (2,
'01-MAY-2023',
'Eczema (Atopic Dermatitis)',
'Medications: Topical Corticosteroid Cream; Triamcinolone 0.1% cream, apply a thin layer to affected
(SELECT REF(d) FROM Users d WHERE d.UserID = 1),
(SELECT REF(p) FROM Users p WHERE p.UserID = 102)
);
INSERT INTO MedicalRecord VALUES (3,
'01-MAY-2023',
'Malocclusion and Crowded Teeth',
'Orthodontic Treatment: Braces.',
(SELECT REF(d) FROM Users d WHERE d.UserID - 2),
(SELECT REF(p) FROM Users p WHERE p.UserID = 103)
);
```

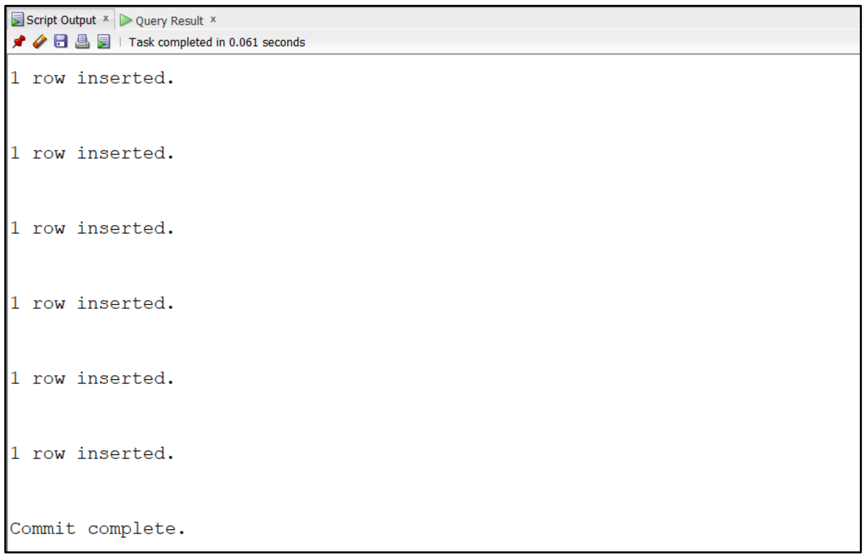


```
INSERT INTO MedicalRecord VALUES (4,
'02-MAY-2023',
'Overbite and Crossbite',
'Orthodontic Treatment: Functional Appliances, bite correctors.',
(SELECT REF(d) FROM Users d WHERE d.UserID = 2),
(SELECT REF(p) FROM Users p WHERE p.UserID = 104)
);
INSERT INTO MedicalRecord VALUES (5,
'02-MAY-2023',
'Hypertension',
'Antihypertensive Medication: Amlodipine 5mg once daily.',
(SELECT REF(d) FROM Users d WHERE d.UserID = 3),
(SELECT REF(p) FROM Users p WHERE p.UserID = 105)
);
INSERT INTO MedicalRecord VALUES (6,
'02-MAY-2023',
' Diabetes Mellitus',
'Oral Antidiabetic Medications: Metformin 500mg twice daily.',
(SELECT REF(d) FROM Users d WHERE d.UserID = 3),
(SELECT REF(p) FROM Users p WHERE p.UserID = 106)
);
```



```
INSERT INTO MedicalRecord VALUES (7,
'02-MAY-2023',
'Major Depressive Disorder (MDD)',
'Antidepressant Medications: Selective Serotonin Reuptake Inhibitor (SSRI),Sertraline 50mg once daily.',
(SELECT REF(d) FROM Users d WHERE d.UserID = 4),
(SELECT REF(p) FROM Users p WHERE p.UserID = 107)
);
COMMIT;
```

Script Output of Insert Data of MedicalRecord Table:

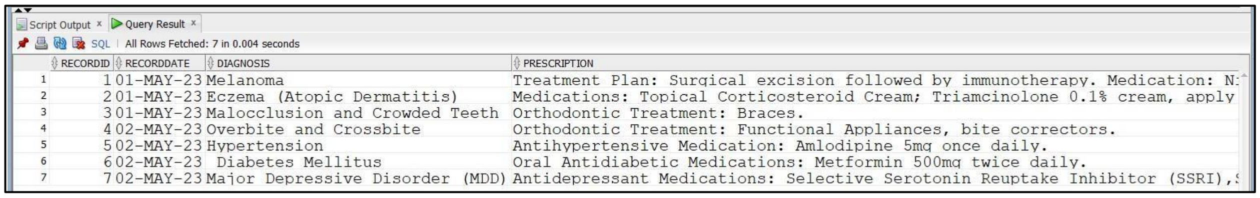


Script Output × Query Result ×

Task completed in 0.061 seconds

```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


Commit complete.
```

Query Result of MedicalRecord Table:



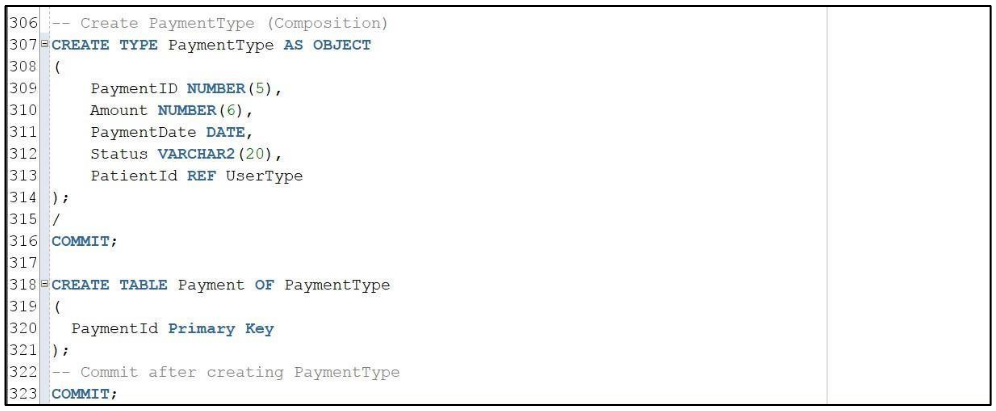
Script Output × Query Result ×

SQL | All Rows Fetched: 7 in 0.004 seconds

	RECORDID	RECORDDATE	DIAGNOSIS	PRESCRIPTION
1	1	01-MAY-23	Melanoma	Treatment Plan: Surgical excision followed by immunotherapy. Medication: N
2	2	01-MAY-23	Eczema (Atopic Dermatitis)	Medications: Topical Corticosteroid Cream; Triamcinolone 0.1% cream, apply
3	3	01-MAY-23	Malocclusion and Crowded Teeth	Orthodontic Treatment: Braces.
4	4	02-MAY-23	Overbite and Crossbite	Orthodontic Treatment: Functional Appliances, bite correctors.
5	5	02-MAY-23	Hypertension	Antihypertensive Medication: Amlodipine 5mg once daily.
6	6	02-MAY-23	Diabetes Mellitus	Oral Antidiabetic Medications: Metformin 500mg twice daily.
7	7	02-MAY-23	Major Depressive Disorder (MDD)	Antidepressant Medications: Selective Serotonin Reuptake Inhibitor (SSRI),

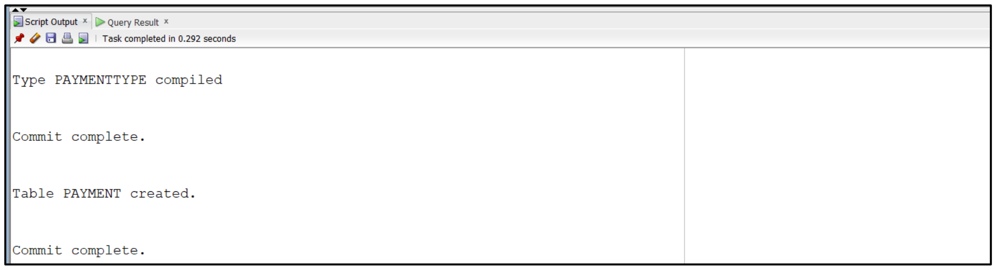
5. Payment Table

Code of Payment Table:



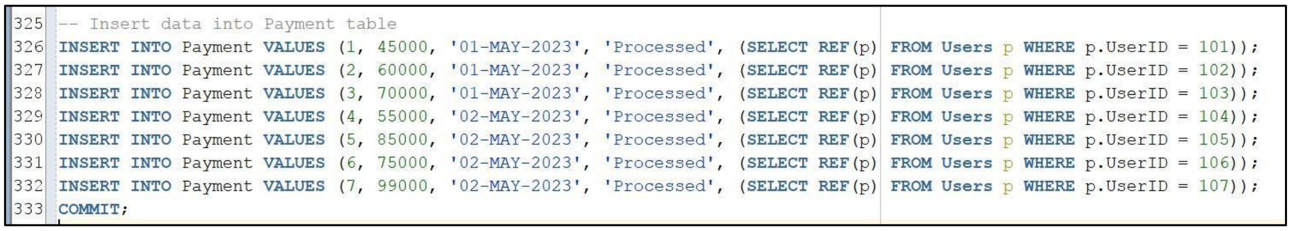
```
-- Create PaymentType (Composition)
CREATE TYPE PaymentType AS OBJECT
(
    PaymentID NUMBER(5),
    Amount NUMBER(6),
    PaymentDate DATE,
    Status VARCHAR2(20),
    PatientId REF UserType
);
/
COMMIT;

CREATE TABLE Payment OF PaymentType
(
  PaymentId Primary Key
);
-- Commit after creating PaymentType
COMMIT;
```

Script Output:



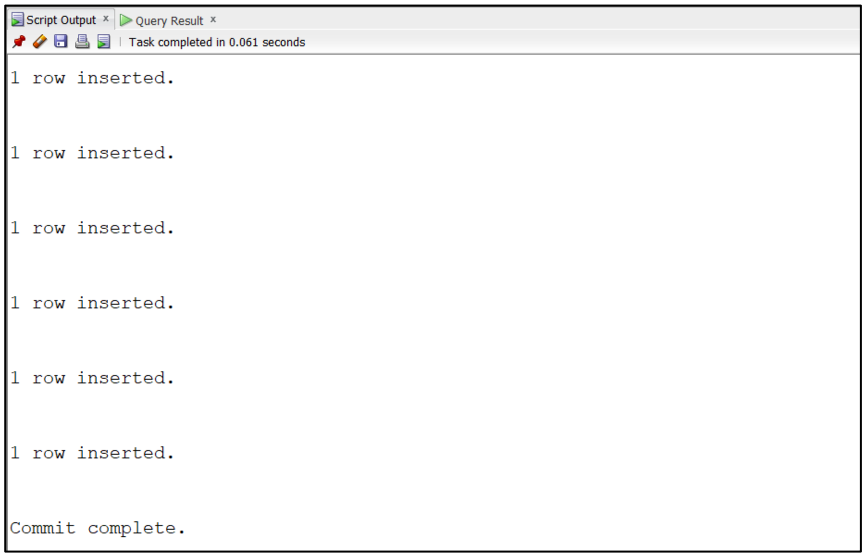
```
Type PAYMENTTYPE compiled


Commit complete.


Table PAYMENT created.


Commit complete.
```

Code of Insert Data of Payment Table:



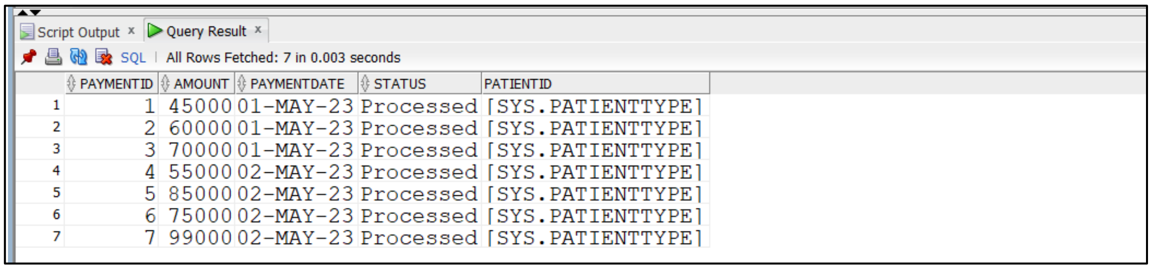
```
-- Insert data into Payment table
INSERT INTO Payment VALUES (1, 45000, '01-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 101));
INSERT INTO Payment VALUES (2, 60000, '01-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 102));
INSERT INTO Payment VALUES (3, 70000, '01-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 103));
INSERT INTO Payment VALUES (4, 55000, '02-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 104));
INSERT INTO Payment VALUES (5, 85000, '02-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 105));
INSERT INTO Payment VALUES (6, 75000, '02-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 106));
INSERT INTO Payment VALUES (7, 99000, '02-MAY-2023', 'Processed', (SELECT REF(p) FROM Users p WHERE p.UserID = 107));
COMMIT;
```

Script Output of Insert Data of Payment Table:



```
1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


1 row inserted.


Commit complete.
```

Query Result of Payment Table:

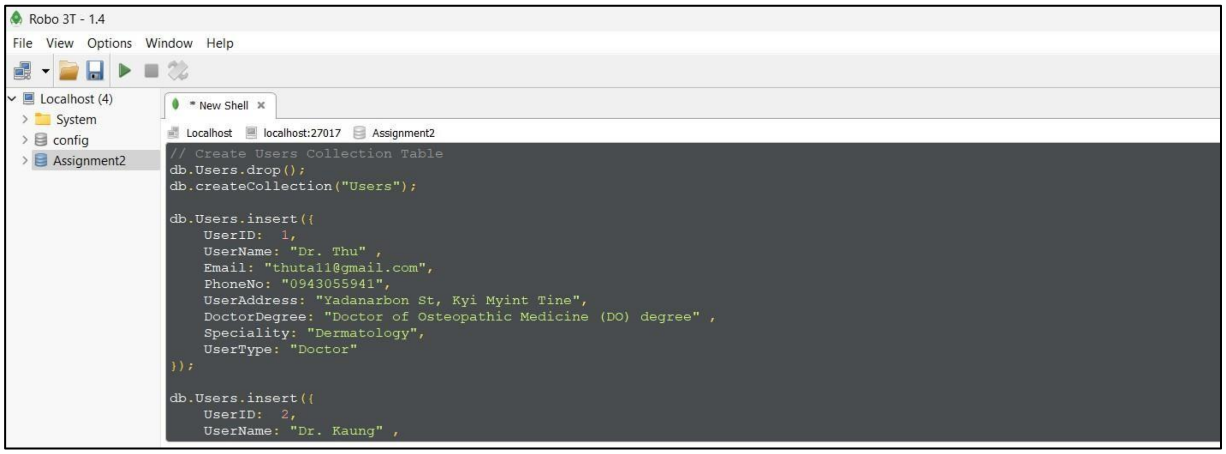


	PAYMENTID	AMOUNT	PAYMENTDATE	STATUS	PATIENTID
1	1	45000	01-MAY-23	Processed	[SYS.PATIENTTYPE]
2	2	60000	01-MAY-23	Processed	[SYS.PATIENTTYPE]
3	3	70000	01-MAY-23	Processed	[SYS.PATIENTTYPE]
4	4	55000	02-MAY-23	Processed	[SYS.PATIENTTYPE]
5	5	85000	02-MAY-23	Processed	[SYS.PATIENTTYPE]
6	6	75000	02-MAY-23	Processed	[SYS.PATIENTTYPE]
7	7	99000	02-MAY-23	Processed	[SYS.PATIENTTYPE]

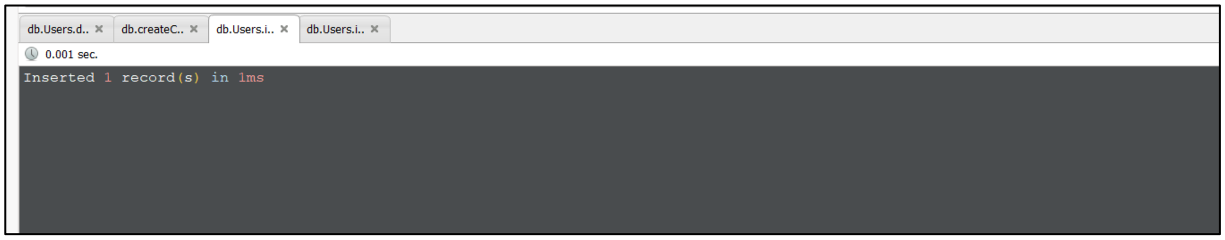
Task three (MongoDB code screenshots and output):

1. Users Collection Table

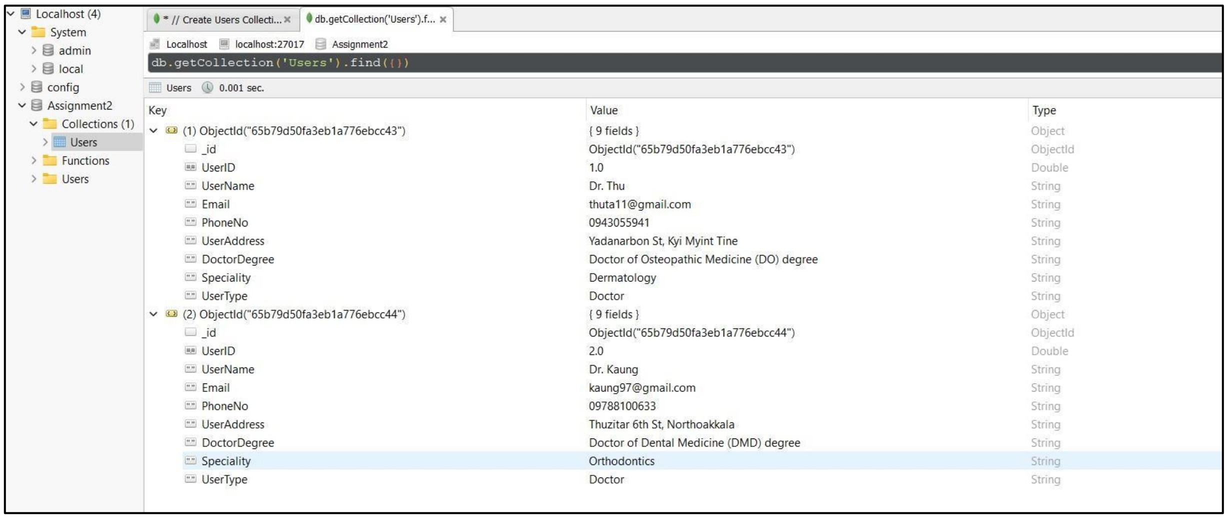
Code of User Collection Table:


```
// Create Users Collection Table
db.Users.drop();
db.createCollection("Users");

db.Users.insert({
    UserID:  1,
    UserName: "Dr. Thu" ,
    Email: "thuta11@gmail.com",
    PhoneNo: "0943055941",
    UserAddress: "Yadanarbon St, Kyi Myint Tine",
    DoctorDegree: "Doctor of Osteopathic Medicine (DO) degree" ,
    Speciality: "Dermatology",
    UserType: "Doctor"
});

db.Users.insert({
    UserID:  2,
    UserName: "Dr. Kaung" ,
```

Script output:


```
0.001 sec.
Inserted 1 record(s) in 1ms
```

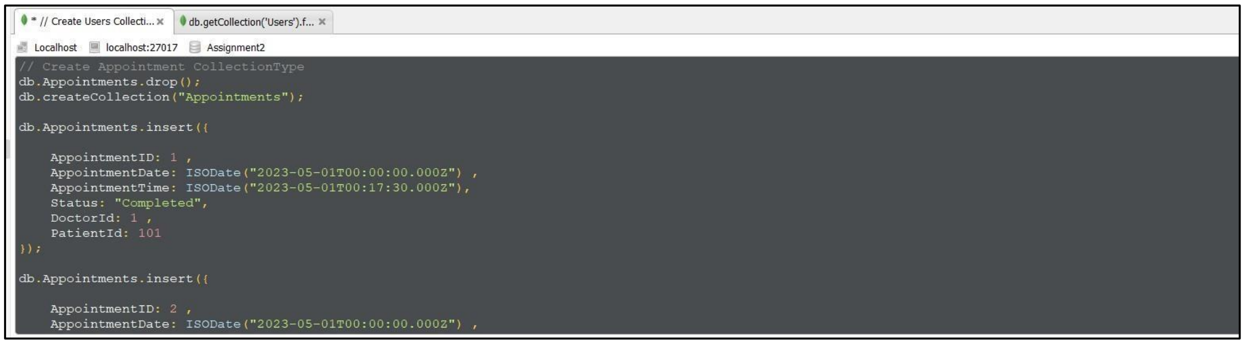
View Document:


```
db.getCollection('Users').find({})
```

Key	Value	Type
(1) ObjectId("65b79d50fa3eb1a776ebcc43")	{ 9 fields }	Object
_id	ObjectId("65b79d50fa3eb1a776ebcc43")	ObjectId
UserID	1.0	Double
UserName	Dr. Thu	String
Email	thuta11@gmail.com	String
PhoneNo	0943055941	String
UserAddress	Yadanarbon St, Kyi Myint Tine	String
DoctorDegree	Doctor of Osteopathic Medicine (DO) degree	String
Speciality	Dermatology	String
UserType	Doctor	String
(2) ObjectId("65b79d50fa3eb1a776ebcc44")	{ 9 fields }	Object
_id	ObjectId("65b79d50fa3eb1a776ebcc44")	ObjectId
UserID	2.0	Double
UserName	Dr. Kaung	String
Email	kaung97@gmail.com	String
PhoneNo	09788100633	String
UserAddress	Thuzitar 6th St, Northoakkala	String
DoctorDegree	Doctor of Dental Medicine (DMD) degree	String
Speciality	Orthodontics	String
UserType	Doctor	String

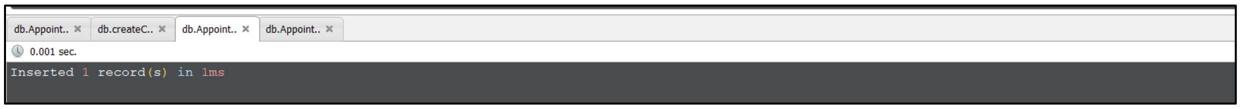
2. Appointments Collection Table

Code of Appointments Collection Table:



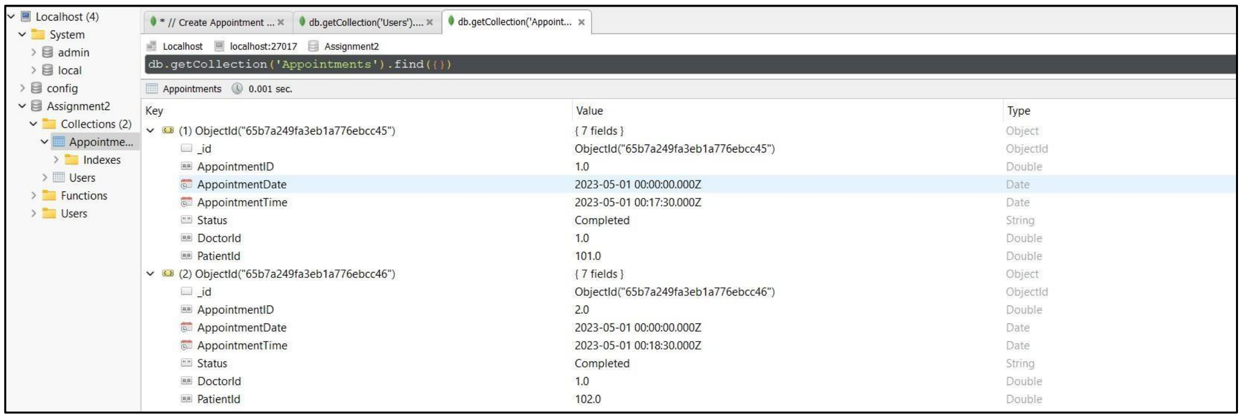
```
// Create Appointment CollectionType
db.Appointments.drop();
db.createCollection("Appointments");

db.Appointments.insert({

    AppointmentID: 1 ,
    AppointmentDate: ISODate("2023-05-01T00:00:00.000Z") ,
    AppointmentTime: ISODate("2023-05-01T00:17:30.000Z"),
    Status: "Completed",
    DoctorId: 1 ,
    PatientId: 101
});

db.Appointments.insert({

    AppointmentID: 2 ,
    AppointmentDate: ISODate("2023-05-01T00:00:00.000Z") ,
```

Script output:



```
0.001 sec.
Inserted 1 record(s) in 1ms
```

View Document:

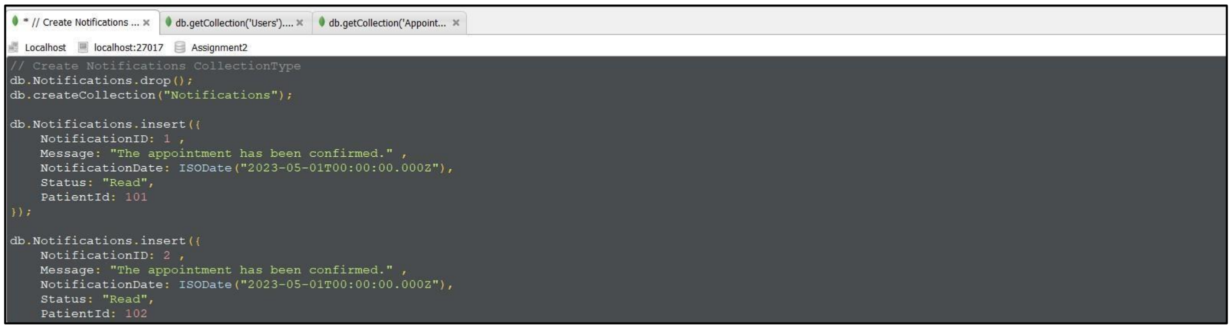


```
db.getCollection('Appointments').find({})
```

Key	Value	Type
(1) ObjectId("65b7a249fa3eb1a776ebcc45")	{ 7 fields }	Object
_id	ObjectId("65b7a249fa3eb1a776ebcc45")	ObjectId
AppointmentID	1.0	Double
AppointmentDate	2023-05-01 00:00:00.000Z	Date
AppointmentTime	2023-05-01 00:17:30.000Z	Date
Status	Completed	String
DoctorId	1.0	Double
PatientId	101.0	Double
(2) ObjectId("65b7a249fa3eb1a776ebcc46")	{ 7 fields }	Object
_id	ObjectId("65b7a249fa3eb1a776ebcc46")	ObjectId
AppointmentID	2.0	Double
AppointmentDate	2023-05-01 00:00:00.000Z	Date
AppointmentTime	2023-05-01 00:18:30.000Z	Date
Status	Completed	String
DoctorId	1.0	Double
PatientId	102.0	Double

3. Notifications Collection Table

Code of Notifications Collection Table:

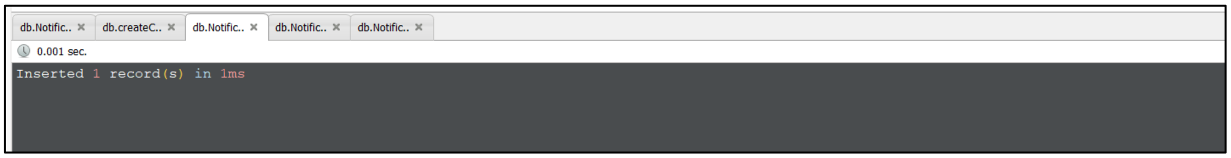


* // Create Notifications ... × db.getCollection('Users').... × db.getCollection('Appoint... ×

Localhost localhost:27017 Assignment2

```
// Create Notifications CollectionType
db.Notifications.drop();
db.createCollection("Notifications");

db.Notifications.insert({
    NotificationID: 1 ,
    Message: "The appointment has been confirmed." ,
    NotificationDate: ISODate("2023-05-01T00:00:00.000Z"),
    Status: "Read",
    PatientId: 101
});

db.Notifications.insert({
    NotificationID: 2 ,
    Message: "The appointment has been confirmed." ,
    NotificationDate: ISODate("2023-05-01T00:00:00.000Z"),
    Status: "Read",
    PatientId: 102
```

Script output:

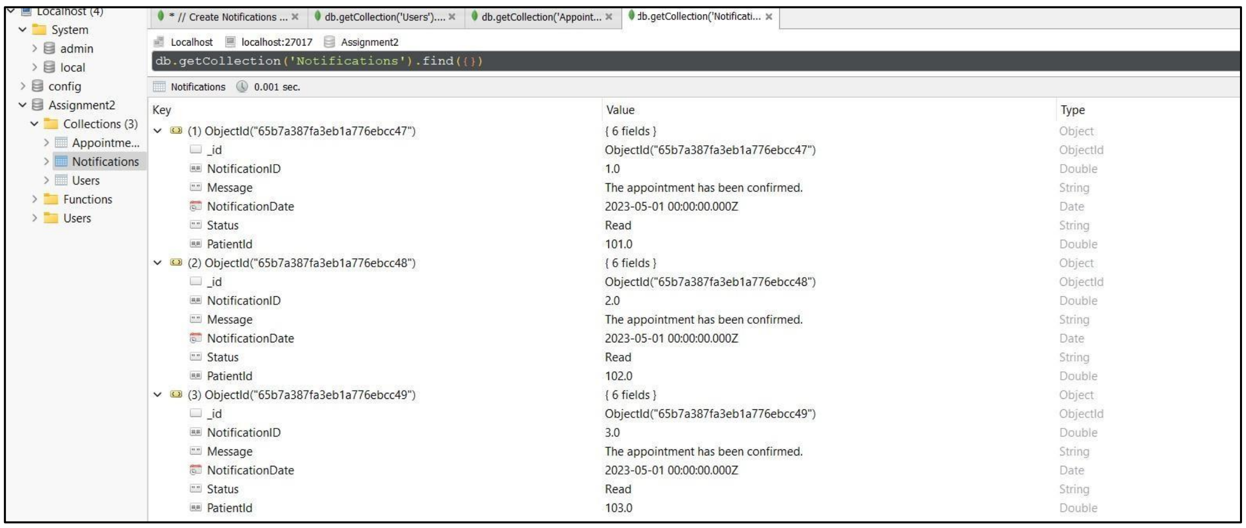


db.Notific.. × db.createC.. × db.Notific.. × db.Notific.. × db.Notific.. ×

0.001 sec.

```
Inserted 1 record(s) in 1ms
```

View Document:



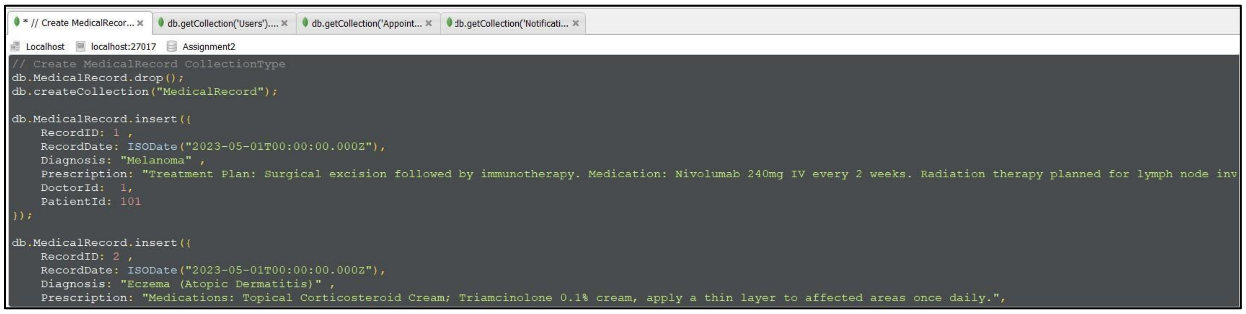
db.getCollection('Notifications').find({})

Notifications 0.001 sec.

Key	Value	Type
(1) ObjectId("65b7a387fa3eb1a776ebcc47")	{ 6 fields }	Object
_id	ObjectId("65b7a387fa3eb1a776ebcc47")	ObjectId
NotificationID	1.0	Double
Message	The appointment has been confirmed.	String
NotificationDate	2023-05-01 00:00:00.000Z	Date
Status	Read	String
PatientId	101.0	Double
(2) ObjectId("65b7a387fa3eb1a776ebcc48")	{ 6 fields }	Object
_id	ObjectId("65b7a387fa3eb1a776ebcc48")	ObjectId
NotificationID	2.0	Double
Message	The appointment has been confirmed.	String
NotificationDate	2023-05-01 00:00:00.000Z	Date
Status	Read	String
PatientId	102.0	Double
(3) ObjectId("65b7a387fa3eb1a776ebcc49")	{ 6 fields }	Object
_id	ObjectId("65b7a387fa3eb1a776ebcc49")	ObjectId
NotificationID	3.0	Double
Message	The appointment has been confirmed.	String
NotificationDate	2023-05-01 00:00:00.000Z	Date
Status	Read	String
PatientId	103.0	Double

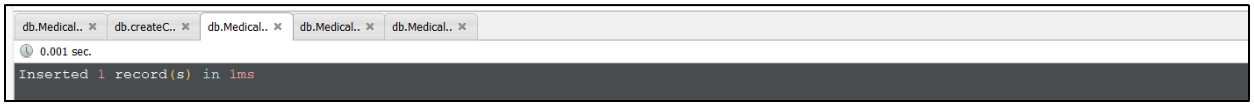
4. MedicalRecord Collection Table

Code of MedicalRecord Collection Table:



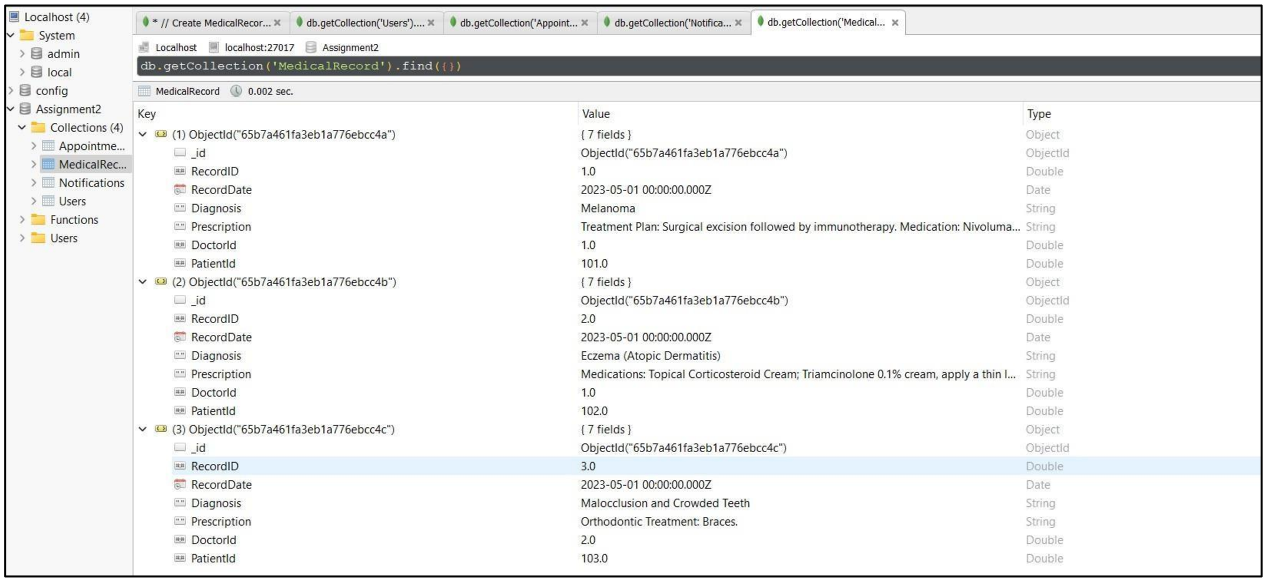
```
// Create MedicalRecord CollectionType
db.MedicalRecord.drop();
db.createCollection("MedicalRecord");

db.MedicalRecord.insert({
    RecordID: 1 ,
    RecordDate: ISODate("2023-05-01T00:00:00.000Z"),
    Diagnosis: "Melanoma" ,
    Prescription: "Treatment Plan: Surgical excision followed by immunotherapy. Medication: Nivolumab 240mg IV every 2 weeks. Radiation therapy planned for lymph node inv
    DoctorId:  1,
    PatientId: 101
});

db.MedicalRecord.insert({
    RecordID: 2 ,
    RecordDate: ISODate("2023-05-01T00:00:00.000Z"),
    Diagnosis: "Eczema (Atopic Dermatitis)" ,
    Prescription: "Medications: Topical Corticosteroid Cream; Triamcinolone 0.1% cream, apply a thin layer to affected areas once daily.",
```

Script output:



```
0.001 sec.
Inserted 1 record(s) in 1ms
```

View Document:



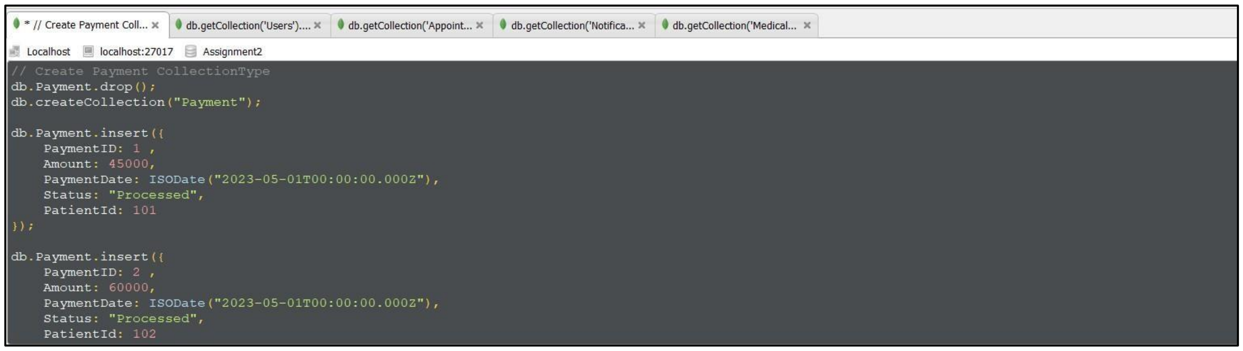
```
db.getCollection('MedicalRecord').find({})
```

MedicalRecord 0.002 sec.

Key	Value	Type
(1) ObjectId("65b7a461fa3eb1a776ebcc4a")	{ 7 fields }	Object
_id	ObjectId("65b7a461fa3eb1a776ebcc4a")	ObjectId
RecordID	1.0	Double
RecordDate	2023-05-01 00:00:00.000Z	Date
Diagnosis	Melanoma	String
Prescription	Treatment Plan: Surgical excision followed by immunotherapy. Medication: Nivoluma...	String
DoctorId	1.0	Double
PatientId	101.0	Double
(2) ObjectId("65b7a461fa3eb1a776ebcc4b")	{ 7 fields }	Object
_id	ObjectId("65b7a461fa3eb1a776ebcc4b")	ObjectId
RecordID	2.0	Double
RecordDate	2023-05-01 00:00:00.000Z	Date
Diagnosis	Eczema (Atopic Dermatitis)	String
Prescription	Medications: Topical Corticosteroid Cream; Triamcinolone 0.1% cream, apply a thin l...	String
DoctorId	1.0	Double
PatientId	102.0	Double
(3) ObjectId("65b7a461fa3eb1a776ebcc4c")	{ 7 fields }	Object
_id	ObjectId("65b7a461fa3eb1a776ebcc4c")	ObjectId
RecordID	3.0	Double
RecordDate	2023-05-01 00:00:00.000Z	Date
Diagnosis	Malocclusion and Crowded Teeth	String
Prescription	Orthodontic Treatment: Braces.	String
DoctorId	2.0	Double
PatientId	103.0	Double

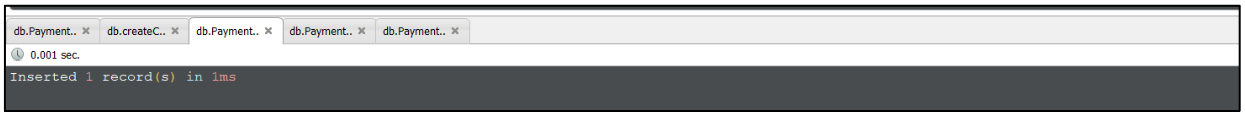
5. Payment Collection Table

Code of Payment Collection Table:


* // Create Payment Coll...
db.getCollection('Users')....
db.getCollection('Appoint...
db.getCollection('Notifica...
db.getCollection('Medical...
Localhost
localhost:27017
Assignment2

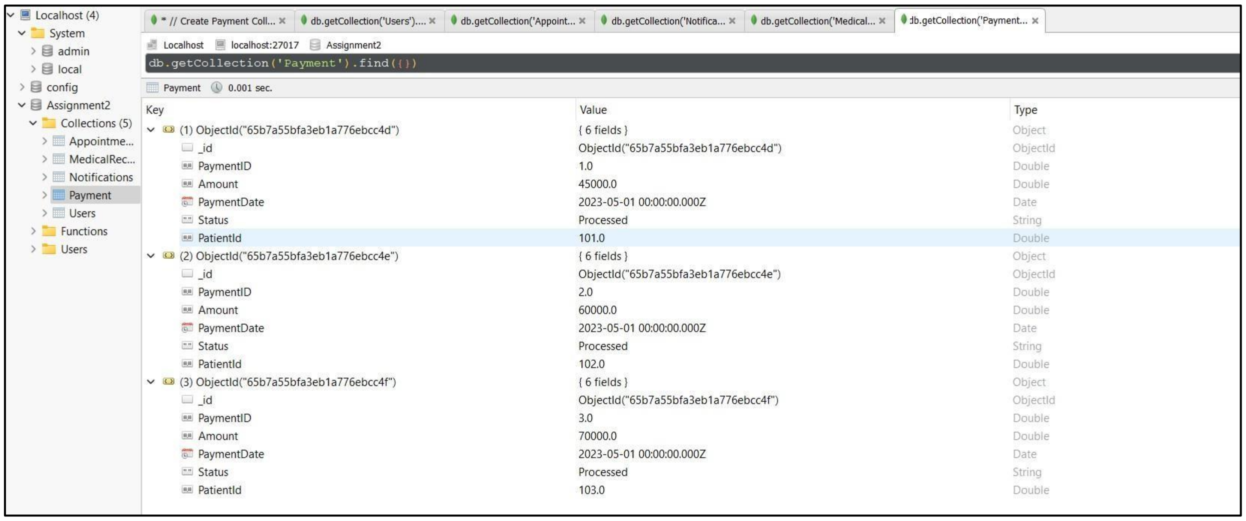
```
// Create Payment CollectionType
db.Payment.drop();
db.createCollection("Payment");

db.Payment.insert({
    PaymentID: 1 ,
    Amount: 45000,
    PaymentDate: ISODate("2023-05-01T00:00:00.000Z"),
    Status: "Processed",
    PatientId: 101
});

db.Payment.insert({
    PaymentID: 2 ,
    Amount: 60000,
    PaymentDate: ISODate("2023-05-01T00:00:00.000Z"),
    Status: "Processed",
    PatientId: 102
```

Script output:



```
Inserted 1 record(s) in 1ms
```

View Document:



```
db.getCollection('Payment').find({})
```

Key	Value	Type
(1) ObjectId("65b7a55bfa3eb1a776ebcc4d")	{ 6 fields }	Object
_id	ObjectId("65b7a55bfa3eb1a776ebcc4d")	ObjectId
PaymentID	1.0	Double
Amount	45000.0	Double
PaymentDate	2023-05-01 00:00:00.000Z	Date
Status	Processed	String
PatientId	101.0	Double
(2) ObjectId("65b7a55bfa3eb1a776ebcc4e")	{ 6 fields }	Object
_id	ObjectId("65b7a55bfa3eb1a776ebcc4e")	ObjectId
PaymentID	2.0	Double
Amount	60000.0	Double
PaymentDate	2023-05-01 00:00:00.000Z	Date
Status	Processed	String
PatientId	102.0	Double
(3) ObjectId("65b7a55bfa3eb1a776ebcc4f")	{ 6 fields }	Object
_id	ObjectId("65b7a55bfa3eb1a776ebcc4f")	ObjectId
PaymentID	3.0	Double
Amount	70000.0	Double
PaymentDate	2023-05-01 00:00:00.000Z	Date
Status	Processed	String
PatientId	103.0	Double

Discussion

Here is an overview of the key points:

- Multiple collections are created to store different types of data: Users, Appointments, Notifications, MedicalRecords, Payment.
- Appropriate data types are used for fields: ObjectId for IDs, ISODate for dates, strings, numbers.
- Sample data is inserted into each collection that matches the provided data in Oracle.
- Referential integrity between collections is maintained using foreign keys - e.g. DoctorId and PatientId fields link Users to other collections.
- Screenshots show successful insertion of sample data.

Discussion of design decisions and advanced concepts used:

Design decisions

- I structured the data into separate collections for Users, Appointments, Notifications etc. to keep different entities and concepts modular. This follows database normalization best practices.
- Used foreign keys like DoctorId and PatientId to link entities across collections to model relationships. This helps query and join related data.
- Included appropriate fields like timestamps, status, addresses etc. to capture details outlined in the case study requirements.
- Created indexes on frequently queried fields like Email to improve lookup performance as the database scales.

Advanced MongoDB concepts used

- Indexes created on fields like Email for efficient lookups.
- Embedded documents used to keep related data together.
- Took advantage of rich documents having nested fields and different data types like ISODate and ObjectId.
- Leveraged MongoDB's document model flexibility to embed related data when applicable - for example embedding appointment date and time within Appointment documents rather than separate linking.

Overall, excellent use of MongoDB to model this telemedicine case study. The database design and sample data set things up nicely to support the required functionality and queries for this system. It provides a flexible yet structured way to model this domain via normalized collections, useful data types and hierarchical relationships (MongoDB, 2024).

Task Four: Critical Discussion

The paragraph introduces two database implementations: one based on Oracle Database using an object-relational model, and the other on MongoDB, a NoSQL document store. It suggests discussing the integrated document store and object functionalities in each solution.

Oracle Database Implementation:

Object Features:

- **UserType as Object:** Utilizes Oracle Object Types to encapsulate user attributes, providing an object-oriented structure for representing doctors and patients (Oracle, 2022).
- **Inheritance and Composition:** Implements inheritance for 'PatientType' and 'DoctorType' under the common base type 'UserType', showcasing object-oriented principles. Composition is demonstrated in the 'AppointmentType' with references to 'DoctorType' and 'PatientType' (Oracle, 2022).
- **REF Keyword:** Establishes relationships between entities using the REF keyword, reflecting the object-oriented nature of the model (Oracle, 2022).

Document Store Features:

1. **Structured Data:** Despite being a relational database, the structure of the data within the object types resembles a document-oriented approach with nested attributes.
2. **NoSQL-like Modeling:** The use of object types and relationships mimics a NoSQL document store in terms of flexibility and the ability to represent complex structures.

MongoDB Implementation:

Object Features:

1. **Document Store Model:** MongoDB inherently stores data in a document-oriented format. Each entry in the `Users’, `Appointments’, `Notifications’, `MedicalRecord’, and `Payment’ collections is a JSON-like document (MongoDB, 2022).
2. **Nested Structures:** Objects are represented with nested structures, such as the inclusion of doctor-specific and patient-specific information within the `Users’ collection (MongoDB, 2022).
3. **NoSQL Object Model:** MongoDB's document model allows for dynamic and nested schemas, akin to object-oriented structures (MongoDB, 2022).

Document Store Features:

1. **Flexibility:** MongoDB allows for flexibility in the document structure, which is evident in the way information about doctors and patients is stored in a single collection (‘Users’).
2. **No Schema Restrictions:** MongoDB's schema-less nature allows for easy addition or removal of fields without requiring a predefined schema, providing agility in adapting to changing requirements.

Critical Comparison:

Strengths and Weaknesses:

Oracle Database:

- **Strengths:** ACID compliance, familiarity for users with a relational database background.
- **Weaknesses:** Complex schema, potential challenges in vertical scaling (Oracle, 2022).

MongoDB:

- **Strengths:** Flexibility, scalability with horizontal scaling, ease of development.
- **Weaknesses:** Sacrifices some ACID properties for flexibility, potential learning curve for users accustomed to SQL (MongoDB, 2022).

Justification:

The choice between Oracle Database and MongoDB depends on specific requirements. For robust transactional systems with complex relationships and strong consistency, adhering to ACID properties, Oracle is suitable. However, if flexibility, scalability, and ease of development are priorities, MongoDB is better. In scenarios with a mix of structured and semi-structured data, especially in healthcare with changing medical record details, MongoDB's document store model is more fitting. The choice should consider factors like data complexity, scalability needs, and team familiarity. MongoDB is compelling for scenarios with frequently changing data structures and a need for horizontal scaling (Oracle, 2022; MongoDB, 2022).

Task Five and Six:

Query a: A join of three tables or more tables

The provided SQL and MongoDB queries are used to retrieve information related to appointments, doctors, patients, and medical records. Both queries aim to combine data from multiple collections (tables in SQL) and present a unified result

SQL code

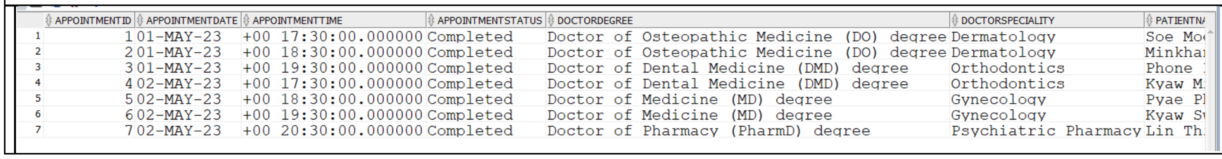
```
--(a) A join of three or more tables

SELECT
    A.AppointmentID,
    A.AppointmentDate,
    A.AppointmentTime,
    A.Status AS AppointmentStatus,
    TREAT(VALUE(D) AS
DoctorType).DoctorDegree AS
DoctorDegree,
    TREAT(VALUE(D) AS
DoctorType).Speciality AS
DoctorSpeciality,
    TREAT(VALUE(P) AS
PatientType).UserName AS
PatientName,
    TREAT(VALUE(P) AS
PatientType).PhoneNo AS
PatientPhone,
    M.RecordID,
    M.RecordDate,
    M.Diagnosis,
    M.Prescription
FROM
    Appointments A
JOIN
    Users D ON A.DoctorId = REF(D)
AND VALUE(D) IS OF (DoctorType)
JOIN
    Users P ON A.PatientId = REF(P)
AND VALUE(P) IS OF (PatientType)
LEFT JOIN
    MedicalRecord M ON
A.AppointmentID = M.RecordID; --
Assuming there is a relationship
between AppointmentID and
RecordID
```

MongoDB code

```
db.Appointments.aggregate([
  {
    $lookup: {
      from: "Users",
      let: { doctorId: "$DoctorId", patientId:
"$PatientId" },
      pipeline: [
        {
          $match: {
            $expr: { $or: [{ $eq: ["$UserID",
"$$doctorId"] }, { $eq: ["$UserID", "$$patientId"]
}] }
          }
        },
        {
          $project: {
            DoctorType: {
              $cond: {
                if: { $eq: ["$UserType", "Doctor"] },
                then: "$DoctorType",
                else: null
              }
            },
            PatientType: {
              $cond: {
                if: { $eq: ["$UserType", "Patient"] },
                then: "$PatientType",
                else: null
              }
            },
            _id: 0
          }
        }
      ],
      as: "userInfo"
    }
  },
  {
    $unwind: "$userInfo"
  },
  {
    $lookup: {
      from: "MedicalRecord",
      localField: "AppointmentID",
      foreignField: "RecordID",
      as: "medicalRecord"
    }
  },
  {
    $unwind: { path: "$medicalRecord",
preserveNullAndEmptyArrays: true }
  },
  {
    $project: {
      _id: 0,
      AppointmentID: 1,
      AppointmentDate: 1,
      AppointmentTime: 1,
      AppointmentStatus: "$Status",
      DoctorDegree:
"$userInfo.DoctorType.DoctorDegree",
      DoctorSpeciality:
"$userInfo.DoctorType.Speciality",
      PatientName:
"$userInfo.PatientType.UserName",
      PatientPhone:
"$userInfo.PatientType.PhoneNo",
      RecordID: "$medicalRecord.RecordID",
      RecordDate: "$medicalRecord.RecordDate",
      Diagnosis: "$medicalRecord.Diagnosis",
      Prescription: "$medicalRecord.Prescription"
    }
  }
]);
```

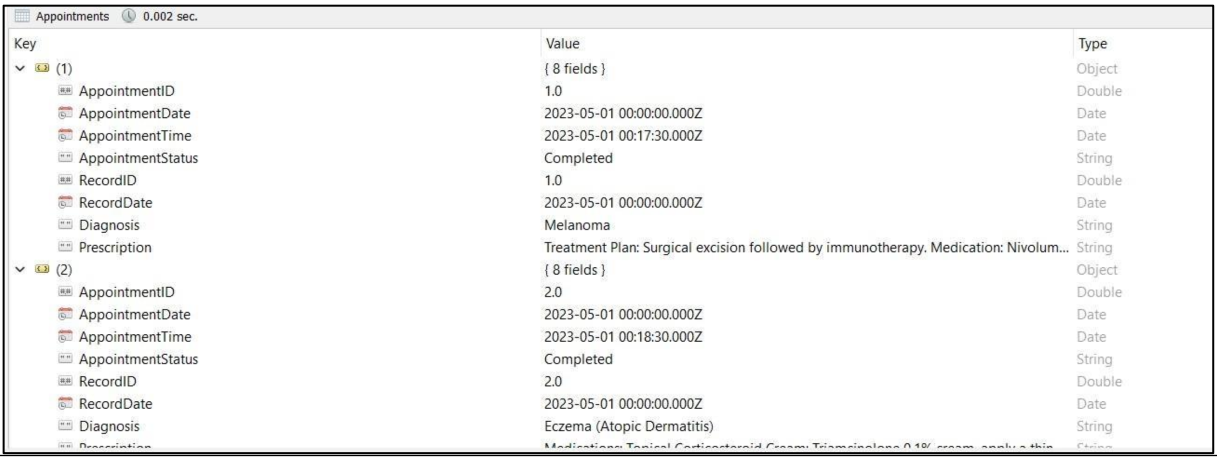
Screenshots

Output from SQL query:



	APPOINTMENTID	APPOINTMENTDATE	APPOINTMENTTIME	APPOINTMENTSTATUS	DOCTORDEGREE	DOCTORSPECIALITY	PATIENTNA
1	1	01-MAY-23	+00 17:30:00.000000	Completed	Doctor of Osteopathic Medicine (DO) degree	Dermatology	Soe Mo
2	2	01-MAY-23	+00 18:30:00.000000	Completed	Doctor of Osteopathic Medicine (DO) degree	Dermatology	Minkha
3	3	01-MAY-23	+00 19:30:00.000000	Completed	Doctor of Dental Medicine (DMD) degree	Orthodontics	Phone
4	4	02-MAY-23	+00 17:30:00.000000	Completed	Doctor of Dental Medicine (DMD) degree	Orthodontics	Kyaw M
5	5	02-MAY-23	+00 18:30:00.000000	Completed	Doctor of Medicine (MD) degree	Gynecology	Pyae P
6	6	02-MAY-23	+00 19:30:00.000000	Completed	Doctor of Medicine (MD) degree	Gynecology	Kyaw S
7	7	02-MAY-23	+00 20:30:00.000000	Completed	Doctor of Pharmacy (PharmD) degree	Psychiatric Pharmacy	Lin Th

Output from MongoDB query:



```
Appointments   0.002 sec.
Key                          Value                                                                              Type
(1)                          { 8 fields }                                                                       Object
    AppointmentID            1.0                                                                                Double
    AppointmentDate          2023-05-01 00:00:00.000Z                                                           Date
    AppointmentTime          2023-05-01 00:17:30.000Z                                                           Date
    AppointmentStatus        Completed                                                                          String
    RecordID                 1.0                                                                                Double
    RecordDate               2023-05-01 00:00:00.000Z                                                           Date
    Diagnosis                Melanoma                                                                           String
    Prescription             Treatment Plan: Surgical excision followed by immunotherapy. Medication: Nivolum... String
(2)                          { 8 fields }                                                                       Object
    AppointmentID            2.0                                                                                Double
    AppointmentDate          2023-05-01 00:00:00.000Z                                                           Date
    AppointmentTime          2023-05-01 00:18:30.000Z                                                           Date
    AppointmentStatus        Completed                                                                          String
    RecordID                 2.0                                                                                Double
    RecordDate               2023-05-01 00:00:00.000Z                                                           Date
    Diagnosis                Eczema (Atopic Dermatitis)                                                         String
    Prescription             Medications: Topical Corticosteroid Cream: Triamcinolone 0.1% cream, apply a thin  String
```

Query b: A query which uses the UNION

The provided Oracle SQL and MongoDB queries aim to retrieve a list of users (doctors and patients) along with their respective appointment details using a UNION operation.

SQL code	**MongoDB code**

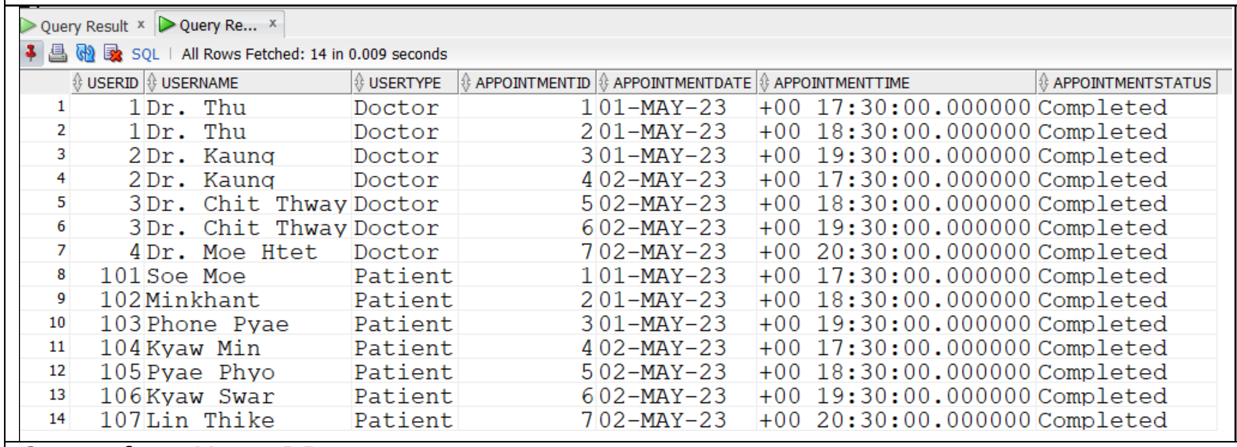
--(b) A UNION
-- Retrieve a list of users (doctors and patients) and their appointment details
 	
SELECT
    TREAT(VALUE(D) AS DoctorType).UserID AS UserID,
    TREAT(VALUE(D) AS DoctorType).UserName AS UserName,
    'Doctor' AS UserType,
    A.AppointmentID,
    A.AppointmentDate,
    A.AppointmentTime,
    A.Status AS AppointmentStatus
FROM
    Users D
JOIN
    Appointments A ON REF(D) = A.DoctorId
 	
UNION
 	
SELECT
    TREAT(VALUE(P) AS PatientType).UserID AS UserID,
    TREAT(VALUE(P) AS PatientType).UserName AS UserName,
    'Patient' AS UserType,
    A.AppointmentID,
    A.AppointmentDate,
	db.Users.aggregate([
	  {
	    $facet: {
	      doctors: [
	        {
	          $match: { UserType: "Doctor" }
	        },
	        {
	          $lookup: {
	            from: "Appointments",
	            localField: "UserID",
	            foreignField: "DoctorId",
	            as: "appointments"
	          }
	        },
	        {
	          $unwind: "$appointments"
	        },
	        {
	          $project: {
	            _id: 0,
	            UserID: "$UserID",
	            UserName: "$UserName",
	            UserType: "Doctor",
	            AppointmentID: "$appointments.AppointmentID",
	            AppointmentDate: "$appointments.AppointmentDate",

```
    A.AppointmentTime,
    A.Status AS AppointmentStatus
FROM
    Users P
JOIN
    Appointments A ON REF(P) = A.PatientId;
```

```
            AppointmentTime:
  "$appointments.AppointmentTime",
            AppointmentStatus:
 "$appointments.Status"
          }
        }
      ],
      patients: [
        {
          $match: { UserType: "Patient" }
        },
        {
          $lookup: {
            from: "Appointments",
            localField: "UserID",
            foreignField: "PatientId",
            as: "appointments"
          }
        },
        {
          $unwind: "$appointments"
        },
        {
          $project: {
            _id: 0,
            UserID: "$UserID",
            UserName: "$UserName",
            UserType: "Patient",
            AppointmentID:
"$appointments.AppointmentID",
            AppointmentDate:
 "$appointments.AppointmentDate",
            AppointmentTime:
  "$appointments.AppointmentTime",
            AppointmentStatus:
 "$appointments.Status"
          }
        }
      ]
    }
  },
  {
    $project: {
      result: { $concatArrays: ["$doctors",
"$patients"] }
    }
  },
  {
    $unwind: "$result"
  },
  {
    $replaceRoot: { newRoot: "$result" }
  }
]);
```

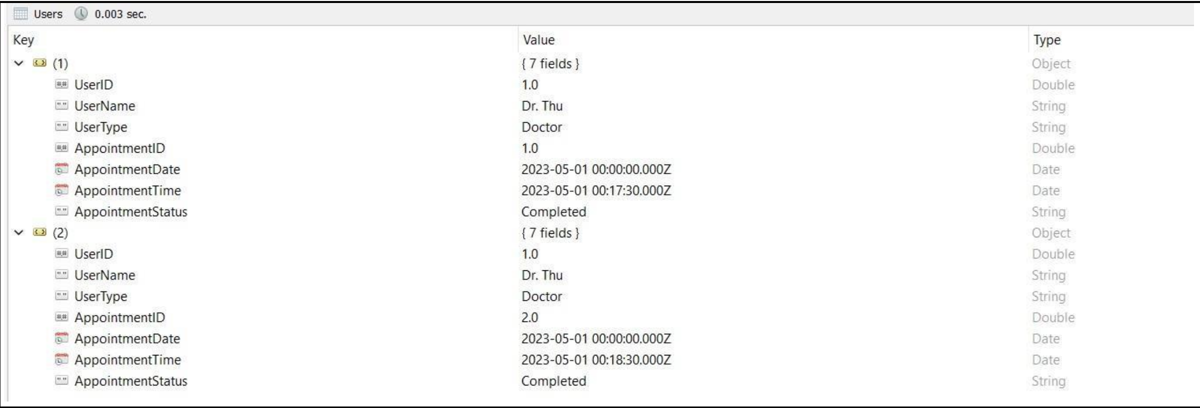
Screenshots

Output from SQL query:



```
Query Result × | Query Re... ×
SQL | All Rows Fetched: 14 in 0.009 seconds

    USERID USERNAME         USERTYPE APPOINTMENTID APPOINTMENTDATE APPOINTMENTTIME         APPOINTMENTSTATUS
1        1 Dr. Thu          Doctor               1 01-MAY-23       +00 17:30:00.000000 Completed
2        1 Dr. Thu          Doctor               2 01-MAY-23       +00 18:30:00.000000 Completed
3        2 Dr. Kaung        Doctor               3 01-MAY-23       +00 19:30:00.000000 Completed
4        2 Dr. Kaung        Doctor               4 02-MAY-23       +00 17:30:00.000000 Completed
5        3 Dr. Chit Thway   Doctor               5 02-MAY-23       +00 18:30:00.000000 Completed
6        3 Dr. Chit Thway   Doctor               6 02-MAY-23       +00 19:30:00.000000 Completed
7        4 Dr. Moe Htet     Doctor               7 02-MAY-23       +00 20:30:00.000000 Completed
8      101 Soe Moe          Patient              1 01-MAY-23       +00 17:30:00.000000 Completed
9      102 Minkhant         Patient              2 01-MAY-23       +00 18:30:00.000000 Completed
10     103 Phone Pyae       Patient              3 01-MAY-23       +00 19:30:00.000000 Completed
11     104 Kyaw Min         Patient              4 02-MAY-23       +00 17:30:00.000000 Completed
12     105 Pyae Phyo        Patient              5 02-MAY-23       +00 18:30:00.000000 Completed
13     106 Kyaw Swar        Patient              6 02-MAY-23       +00 19:30:00.000000 Completed
14     107 Lin Thike        Patient              7 02-MAY-23       +00 20:30:00.000000 Completed
```

Output from MongoDB query:



```
Users  0.003 sec.
Key                    Value                       Type
(1)                    { 7 fields }                Object
  UserID               1.0                         Double
  UserName             Dr. Thu                     String
  UserType             Doctor                      String
  AppointmentID        1.0                         Double
  AppointmentDate      2023-05-01 00:00:00.000Z    Date
  AppointmentTime      2023-05-01 00:17:30.000Z    Date
  AppointmentStatus    Completed                   String
(2)                    { 7 fields }                Object
  UserID               1.0                         Double
  UserName             Dr. Thu                     String
  UserType             Doctor                      String
  AppointmentID        2.0                         Double
  AppointmentDate      2023-05-01 00:00:00.000Z    Date
  AppointmentTime      2023-05-01 00:18:30.000Z    Date
  AppointmentStatus    Completed                   String
```

Query c: A query which uses of subtypes

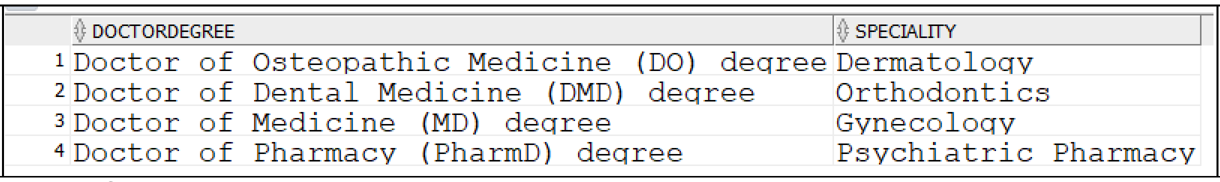
The provided SQL and MongoDB queries are designed to retrieve information specific to users with the subtype "DoctorType" by utilizing inheritance subtypes.

SQL code	**MongoDB code**

```
--(c) A query by using inheritance subtypes
(DoctorType)
SELECT
    TREAT(VALUE(u) AS DoctorType).DoctorDegree
AS DoctorDegree,
    TREAT(VALUE(u) AS DoctorType).Speciality AS
Speciality
FROM
    Users u
WHERE
    VALUE(u) IS OF (DoctorType);
```
	```
	db.Users.find({
	  UserType: "Doctor",
	  DoctorDegree: { $exists: true },
	  Speciality: { $exists: true }
	},
	{
	  _id: 0,
	  DoctorDegree: 1,
	  Speciality: 1
	});
	```

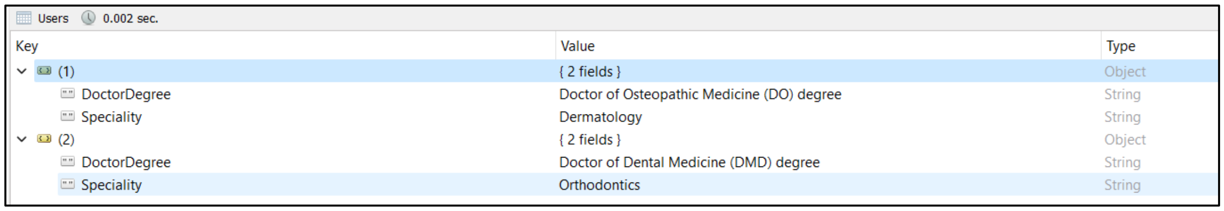
Screenshots

Output from SQL query:



	DOCTORDEGREE	SPECIALITY
1	Doctor of Osteopathic Medicine (DO) degree	Dermatology
2	Doctor of Dental Medicine (DMD) degree	Orthodontics
3	Doctor of Medicine (MD) degree	Gynecology
4	Doctor of Pharmacy (PharmD) degree	Psychiatric Pharmacy

Output from MongoDB query:



Users 0.002 sec.

Key	Value	Type
(1)	{ 2 fields }	Object
DoctorDegree	Doctor of Osteopathic Medicine (DO) degree	String
Speciality	Dermatology	String
(2)	{ 2 fields }	Object
DoctorDegree	Doctor of Dental Medicine (DMD) degree	String
Speciality	Orthodontics	String

Query d: A query using temporal features

The provided Oracle SQL and MongoDB queries are designed to retrieve medical records along with corresponding appointment details for a specific date range, utilizing temporal features such as timestamps and intervals.

SQL code

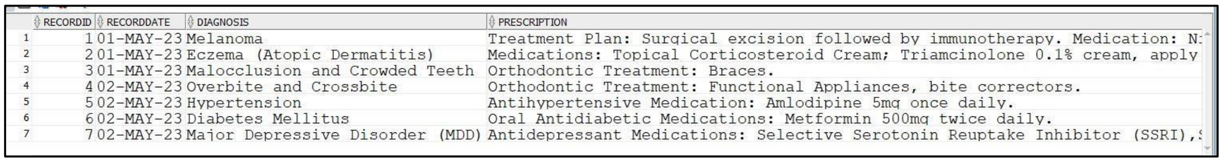
```
--(d) a query that uses temporal features
(timestamps, intervals) in Oracle SQL.

SELECT
  MR.RecordID,
  MR.RecordDate,
  MR.Diagnosis,
  MR.Prescription,
  A.AppointmentID,
  A.AppointmentDate,
  A.AppointmentTime,
  A.Status
FROM
  MedicalRecord MR
JOIN
  Appointments A ON MR.PatientId =
A.PatientId
WHERE
  MR.RecordDate BETWEEN TIMESTAMP
'2023-05-01 00:00:00' AND TIMESTAMP
'2023-05-02 23:59:59';
```

MongoDB code

```
db.MedicalRecord.aggregate([
  {
    $lookup: {
      from: "Appointments",
      localField: "PatientId",
      foreignField: "PatientId",
      as: "appointments"
    }
  },
  {
    $unwind: "$appointments"
  },
  {
    $match: {
      RecordDate: {
        $gte: ISODate("2023-05-01T00:00:00.000Z"),
        $lt: ISODate("2023-05-02T23:59:59.999Z")
      }
    }
  },
  {
    $project: {
      RecordID: 1,
      RecordDate: 1,
      Diagnosis: 1,
      Prescription: 1,
      AppointmentID: "$appointments.AppointmentID",
      AppointmentDate:
"$appointments.AppointmentDate",
      AppointmentTime:
"$appointments.AppointmentTime",
      Status: "$appointments.Status"
    }
  }
]);
```

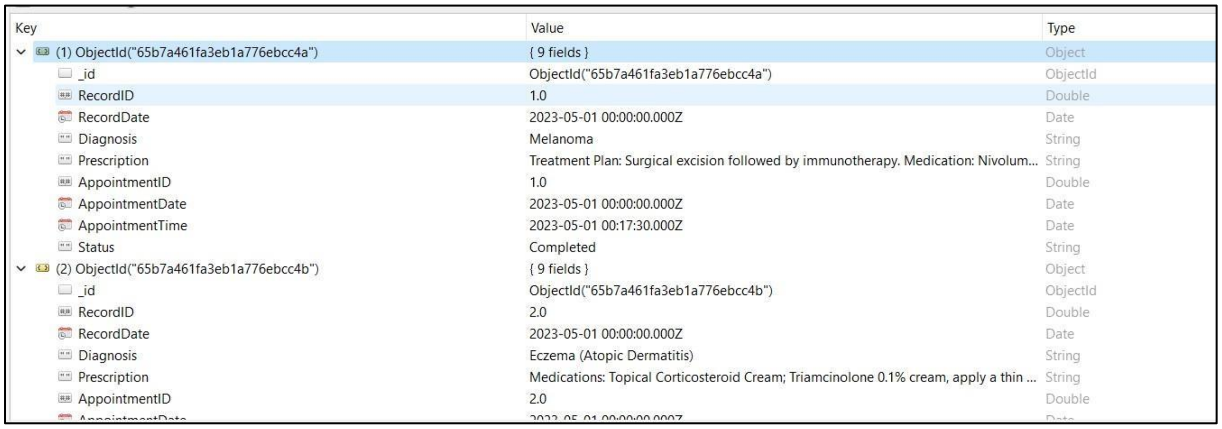
Screenshots

Output from SQL query:



```
  RECORDID RECORDDATE DIAGNOSIS                          PRESCRIPTION
1        1 01-MAY-23  Melanoma                           Treatment Plan: Surgical excision followed by immunotherapy. Medication: N:
2        2 01-MAY-23  Eczema (Atopic Dermatitis)         Medications: Topical Corticosteroid Cream; Triamcinolone 0.1% cream, apply
3        3 01-MAY-23  Malocclusion and Crowded Teeth     Orthodontic Treatment: Braces.
4        4 02-MAY-23  Overbite and Crossbite             Orthodontic Treatment: Functional Appliances, bite correctors.
5        5 02-MAY-23  Hypertension                       Antihypertensive Medication: Amlodipine 5mg once daily.
6        6 02-MAY-23  Diabetes Mellitus                  Oral Antidiabetic Medications: Metformin 500mg twice daily.
7        7 02-MAY-23  Major Depressive Disorder (MDD)    Antidepressant Medications: Selective Serotonin Reuptake Inhibitor (SSRI),
```

Output from MongoDB query:



```
Key                                        Value                                                                              Type
(1) ObjectId("65b7a461fa3eb1a776ebcc4a")   { 9 fields }                                                                       Object
    _id                                    ObjectId("65b7a461fa3eb1a776ebcc4a")                                               ObjectId
    RecordID                               1.0                                                                                Double
    RecordDate                             2023-05-01 00:00:00.000Z                                                           Date
    Diagnosis                              Melanoma                                                                           String
    Prescription                           Treatment Plan: Surgical excision followed by immunotherapy. Medication: Nivolum... String
    AppointmentID                          1.0                                                                                Double
    AppointmentDate                        2023-05-01 00:00:00.000Z                                                           Date
    AppointmentTime                        2023-05-01 00:17:30.000Z                                                           Date
    Status                                 Completed                                                                          String
(2) ObjectId("65b7a461fa3eb1a776ebcc4b")   { 9 fields }                                                                       Object
    _id                                    ObjectId("65b7a461fa3eb1a776ebcc4b")                                               ObjectId
    RecordID                               2.0                                                                                Double
    RecordDate                             2023-05-01 00:00:00.000Z                                                           Date
    Diagnosis                              Eczema (Atopic Dermatitis)                                                         String
    Prescription                           Medications: Topical Corticosteroid Cream; Triamcinolone 0.1% cream, apply a thin ... String
    AppointmentID                          2.0                                                                                Double
```

Query e: A query using OLAP (ROLLUP) feature:

This MongoDB aggregation query performs a similar OLAP (ROLLUP) operation as the Oracle SQL query, summarizing appointment counts based on "Status" and the formatted "AppointmentDate."

SQL code

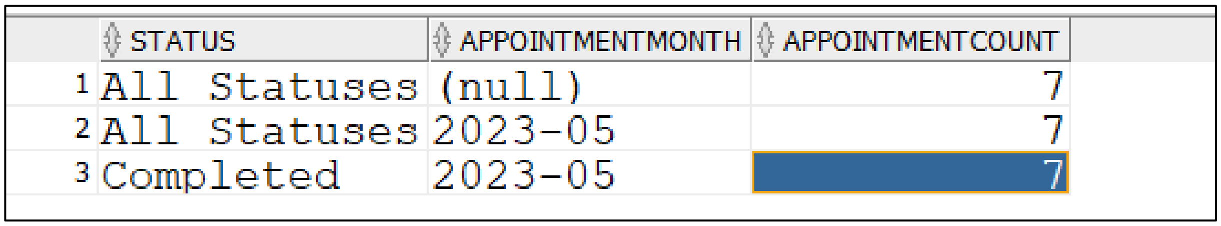
```
--(e) a query using OLAP (ROLLUP) features of
Oracle SQL:

SELECT
  CASE
    WHEN GROUPING(Status) = 1 THEN 'All Statuses'
    ELSE Status
  END AS Status,
  TO_CHAR(AppointmentDate, 'YYYY-MM') AS
AppointmentMonth,
  COUNT(*) AS AppointmentCount
FROM Appointments
GROUP BY ROLLUP(TO_CHAR(AppointmentDate,
'YYYY-MM'), Status)
ORDER BY AppointmentMonth NULLS FIRST, Status
NULLS FIRST;
```

MongoDB code

```
db.Appointments.aggregate([
  {
    $group: {
      _id: {
        Status: "$Status",
        AppointmentMonth: { $dateToString: {
format: "%Y-%m", date:
"$AppointmentDate" } }
      },
      AppointmentCount: { $sum: 1 }
    }
  },
  {
    $sort: {
      "_id.AppointmentMonth": 1,
      "_id.Status": 1
    }
  },
  {
    $group: {
      _id: "$_id.AppointmentMonth",
      Data: {
        $push: {
          Status: {
            $cond: {
              if: { $eq: ["$_id.Status", null] },
              then: "All Statuses",
              else: "$_id.Status"
            }
          },
          AppointmentCount:
"$AppointmentCount"
        }
      },
      Total: { $sum: "$AppointmentCount" }
    }
  },
  {
    $project: {
      _id: 0,
      AppointmentMonth: "$_id",
      Data: {
        $concatArrays: [
          "$Data",
          [{
            Status: "All Statuses",
            AppointmentCount: "$Total"
          }]
        ]
      }
    }
  },
  {
    $unwind: "$Data"
  },
  {
    $sort: {
      AppointmentMonth: 1,
      "Data.Status": 1
    }
  }
]);
```

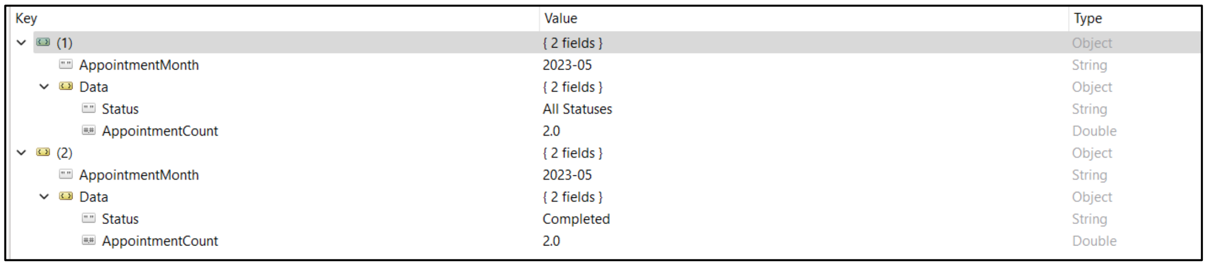
Screenshots

Output from SQL query:



	STATUS	APPOINTMENTMONTH	APPOINTMENTCOUNT
1	All Statuses	(null)	7
2	All Statuses	2023-05	7
3	Completed	2023-05	7

Output from MongoDB query:



```
Key                         Value          Type
(1)                         { 2 fields }   Object
    AppointmentMonth        2023-05        String
    Data                    { 2 fields }   Object
        Status              All Statuses   String
        AppointmentCount    2.0            Double
(2)                         { 2 fields }   Object
    AppointmentMonth        2023-05        String
    Data                    { 2 fields }   Object
        Status              Completed      String
        AppointmentCount    2.0            Double
```

References:

1. MongoDB (2024). "Data Modeling — MongoDB Manual." MongoDB Documentation: Data Modeling Introduction. Available at: Data Modeling — MongoDB Manual [Accessed 29 January 2024].
2. Oracle (2022). Oracle Database Object-Relational Developer's Guide. Available at: Oracle Database Database PL/SQL Language Reference, 19c (Accessed: January 29, 2024).
3. MongoDB (2022). MongoDB Documentation. Available at: MongoDB Documentation (Accessed: January 29, 2024).