```
/* Query 1:
Display all dentists' contact information with their name and DID, sorted by DID */
SELECT DID, Name, Phone, Address
FROM Dentist
ORDER BY DID;

/* Query 2:
Display the cost of each kind of filling service (with their names), SID not needed */
SELECT Name AS 'Filling Service', Cost
FROM Service
WHERE Name LIKE 'Filling%';

/* Query 3:
Insert one new appointment by Sherlock Holmes with Alan To, May 4, 2023, 10:00, 2 hours */
INSERT INTO Appointment VALUES(8, '2023-05-04', '10:00', '02:00', 2, 2);

/* Query 4:
Create a view for the clerks to see appointment information on or after today, including:
Appointment date, start time, end time, names of dentist and patient.
Then, display all information of this view.*/
/* Remarks:
(a) Reference for extracting hours and minutes information from a time value (DATEPART()):
https://learn.microsoft.com/en-us/sql/t-sql/functions/datepart-transact-sql?view=sql-server-ver16
(b) Reference for adding time values together (DATEADD()):
https://learn.microsoft.com/en-us/sql/t-sql/functions/dateadd-transact-sql?view=sql-server-ver16
(c) Reference for obtaining current date value (GETDATE() and CONVERT()):
https://learn.microsoft.com/en-us/sql/t-sql/functions/getdate-transact-sql?view=sql-server-ver16
(d) MySQL uses different function names for adding time and getting current date */
CREATE VIEW Future_appointments AS
SELECT Date, Time as StartTime, DATEADD(hh, DATEPART(hh, Duration), DATEADD(mi,
(DATEPART(mi, Duration)), Time)) AS EndTime, D.Name AS Dentist, P.Name as Patient
FROM (Appointment A INNER JOIN Dentist D ON A.DID = D.DID)
                INNER JOIN Patient P ON A.PID = P.PID
WHERE A.Date > CONVERT(date, GETDATE());

SELECT Date, StartTime, EndTime, Dentist, Patient
FROM Future_appointments;

/* Query 5:
Display the names and phone numbers of patients who have never done any basic checkup */
SELECT Name, Phone
FROM Patient
EXCEPT
```

```
SELECT DISTINCT P.Name, P.Phone
FROM Patient P, Appointment A, Service S, Service_Involved_In_Appointment SA
WHERE P.PID = A.PID
        AND A.AID = SA.AID
   AND S.SID = SA.SID
   AND EXISTS (SELECT * FROM Service WHERE S.Name = 'Basic checkup');

/* Query 6:
Display the total payment made for each appointment, with the AID;
only consider past appointments i.e. appointment before today */
/* Remarks:
(a) Need to replace NULL values with 0:
https://learn.microsoft.com/en-us/sql/t-sql/functions/isnull-transact-sql?view=sql-server-ver16
(b) MySQL uses different function names for converting null value and getting current date */
SELECT PMT.AID, SUM(ISNULL(PMT.PolicyAmount, 0) + ISNULL(PMT.PatientAmount, 0)) AS
TotalPaymentMade
FROM Payment PMT, Appointment A
WHERE A.Date < CONVERT(date, GETDATE())
        AND PMT.AID = A.AID
GROUP BY PMT.AID
UNION
-- Consider appointment with no payment made at all
SELECT AID, ISNULL(NULL, 0) AS TotalPaymentMade
FROM Appointment
WHERE AID NOT IN (SELECT DISTINCT AID FROM Payment)
        AND Date < CONVERT(date, GETDATE())
ORDER BY AID;

/* Query 7:
Display the AID and total cost of each appointment, sort by AID;
only consider past appointments i.e. appointment before today */
/* Remarks:
(a) MySQL uses different function name for getting current date */
SELECT A.AID, SUM(S.Cost) AS TotalCost
FROM Appointment A, Service S, Service_Involved_In_Appointment SA
WHERE A.AID = SA.AID
        AND SA.SID = S.SID
   AND A.Date < CONVERT(date, GETDATE())
GROUP BY A.AID;

/* Query 8:
Display the customers' names and phone numbers who still have outstanding payment,
and the corresponding appointment(s) and amount(s)
```

```
(Note: Only consider past appointments i.e. before today) */
/* Remarks:
(a) MySQL uses different function names for converting null value and getting current date */
SELECT P.Name, P.Phone, A.Date AS AppointmentDate, total.TotalCost -
total_paid.TotalPaymentMade AS OutstandingAmount
FROM (SELECT A.AID, SUM(S.Cost) AS TotalCost
                FROM Appointment A, Service S, Service_Involved_In_Appointment SA
                WHERE A.AID = SA.AID
                        AND SA.SID = S.SID
                        AND A.Date < CONVERT(date, GETDATE())
                GROUP BY A.AID) total,
        (SELECT PMT.AID, SUM(ISNULL(PMT.PolicyAmount, 0) +
ISNULL(PMT.PatientAmount, 0)) AS TotalPaymentMade
                FROM Payment PMT, Appointment A
                WHERE A.Date < CONVERT(date, GETDATE()) AND PMT.AID = A.AID
                GROUP BY PMT.AID
                UNION
                SELECT AID, ISNULL(NULL, 0) AS TotalPaymentMade
                FROM Appointment
                WHERE AID NOT IN (SELECT DISTINCT AID FROM Payment)
                        AND Date < CONVERT(date, GETDATE())) total_paid,
        Appointment A,
   Patient P
WHERE total.TotalCost > total_paid.TotalPaymentMade
        AND A.AID = total.AID
   AND A.AID = total_paid.AID
   AND A.PID = P.PID;

/* Query 9:
Change the previously inserted appointment to 13:00 */
UPDATE Appointment
SET Time = '13:00'
WHERE AID = 8;

/* Query 10:
Display the total spending of each patient and the patient's name, sorted by amount in
descending order
(Note: Only consider past appointments i.e. before today) */
/* Remarks:
(a) The GROUP BY clause can only use the column selected e.g. P.Name, but not others e.g.
P.PID.
Otherwise, MS SQL Server throws error Msg 8120; although this error does not happen in
MySQL.
```

```
While the result here is OKAY, there may be problem when two patients have the same name;
therefore, it may be wiser to display the PID as well to ensure uniqueness of each row. */
SELECT P.Name, SUM(S.Cost) as TotalSpending
FROM Patient P, Appointment A, Service_Involved_In_Appointment SA, Service S
WHERE S.SID = SA.SID
        AND SA.AID = A.AID
   AND A.PID = P.PID
GROUP BY P.Name
ORDER BY TotalSpending DESC;

/* Query 11:
Display the number of appointments each dentist has received, together with their names,
sorted in descending order */
/* Remarks:
(a) MySQL uses different function name for converting null value
(b) The GROUP BY clause can only use D.Name to avoid error Msg 8120 same as the one in
query 10. */
SELECT D.Name, COUNT(A.DID) AS TotalAppointment
FROM Appointment A, Dentist D
WHERE A.DID = D.DID
GROUP BY D.Name
UNION
-- Consider dentists with no appointment received
SELECT Name, ISNULL(NULL, 0)
FROM Dentist
WHERE DID NOT IN (SELECT DISTINCT DID FROM Appointment)
ORDER BY TotalAppointment DESC;

/* Query 12:
Display the amount of revenue generated by each dentist, together with their names, sorted in
descending order */
/* Remarks:
(a) MySQL uses different function name for converting null value
(b) The GROUP BY clause can only use D.Name to avoid error Msg 8120 same as the one in
query 10. */
SELECT D.Name, SUM(S.Cost) AS TotalRevenue
FROM Dentist D, Service S, Appointment A, Service_Involved_In_Appointment SA
WHERE D.DID = A.DID
        AND A.AID = SA.AID
   AND SA.SID = S.SID
GROUP BY D.Name
UNION
-- Consider dentists with no appointment received
```

```
SELECT Name, ISNULL(NULL, 0)
FROM Dentist
WHERE DID NOT IN (SELECT DISTINCT DID FROM Appointment)
ORDER BY TotalRevenue DESC;

/* Query 13:
Display the total spending of each patient, together with their names and joining dates, sorted in
ascending order of joining dates */
/* Remarks:
(a) The GROUP BY clause needs both P.Name and P.DateJoined to avoid error Msg 8120
same as the one in query 10. */
SELECT P.Name, P.DateJoined, SUM(S.Cost) AS TotalSpending
FROM Patient P, Appointment A, Service_Involved_In_Appointment SA, Service S
WHERE P.PID = A.PID
        AND A.AID = SA.AID
   AND SA.SID = S.SID
GROUP BY P.Name, P.DateJoined
ORDER BY P.DateJoined;

/* Query 14:
Insert a new kind of service about teeth whitening, with cost of 1000 */
INSERT INTO Service VALUES(8, 'Teeth whitening', 1000);

/* Query 15:
It turns out that the whitening service is not ready without the specialist dentist;
delete that service */
DELETE from Service
WHERE SID = 8;

/* Query 16:
Display the each dentist's name and her/his total duration of appointment has provided,
sorted by duration in descending order */
/* Remarks:
(a) The GROUP BY clause can only use D.Name to avoid error Msg 8120 same as the one in
query 10.
(b) MySQL uses different function name for converting null value
(c) Without corresponding functions like those in MySQL, it is difficult to display the total duration
of appointments of each dentist in proper format.
As a fallback, each duration value is converted to minutes by using DATEDIFF() with 00:00:00
as a reference point;
then, the total time in minute is obtained by using SUM().
Ref: https://stackoverflow.com/a/9725789/16071474 */
SELECT D.Name, SUM(DATEDIFF(mi, '00:00:00', Duration)) AS TotalTimeInMinute
```

```
FROM Dentist D, Appointment A
WHERE A.DID = D.DID
GROUP BY D.Name
UNION
-- Consider dentists with no appointment received
SELECT Name, ISNULL(NULL, 0)
FROM Dentist
WHERE DID NOT IN (SELECT DISTINCT DID FROM Appointment)
ORDER BY TotalTimeInMinute DESC;

/* Query 17:
Display the total duration of each patient's appointment(s)
sorted by duration in descending order */
/* Remarks:
(a) The GROUP BY clause can only use P.Name to avoid error Msg 8120 same as the one in query 10.
(b) The approach of calculating the total duration is different from that used in MySQL,
with the same reason in query 16. */
SELECT P.Name, SUM(DATEDIFF(mi, '00:00:00', Duration)) AS TotalTimeInMinute
FROM Patient P INNER JOIN Appointment A ON P.PID = A.PID
GROUP BY P.Name
ORDER BY TotalTimeInMinute DESC;

/* Query 18:
Display the latest appointment (including those in future) made by each patient, with their names.
Sort by the dates in ascending order. */
/* Remarks:
(a) The GROUP BY clause can only use P.Name to avoid error Msg 8120 same as the one in query 10. */
SELECT P.Name, MAX(A.Date) AS LatestAppointment
FROM Patient P INNER JOIN Appointment A ON P.PID = A.PID
GROUP BY P.Name
ORDER BY LatestAppointment;

/* Query 19:
Update the phone number of Patient John Stuart Mill to 7789909981 */
UPDATE Patient
SET Phone = '7789909981'
WHERE PID = 3;

/* Query 20:
Display the name, policy information, and sum of related payment amounts of patients who
```

```
have made at least one payment with insurance policy */
/* Remarks:
(a) The GROUP BY clause can only use P.Name to avoid error Msg 8120 same as the one in
query 10. */
SELECT P.Name, PMT.Insurer, PMT.PolicyNo, SUM(PMT.PolicyAmount) AS
TotalPaymentByPolicy
FROM (Appointment A INNER JOIN Payment PMT ON A.AID = PMT.AID)
        INNER JOIN Patient P ON P.PID = A.PID
WHERE PMT.PolicyAmount IS NOT NULL
GROUP BY P.Name, PMT.Insurer, PMT.PolicyNo;
```