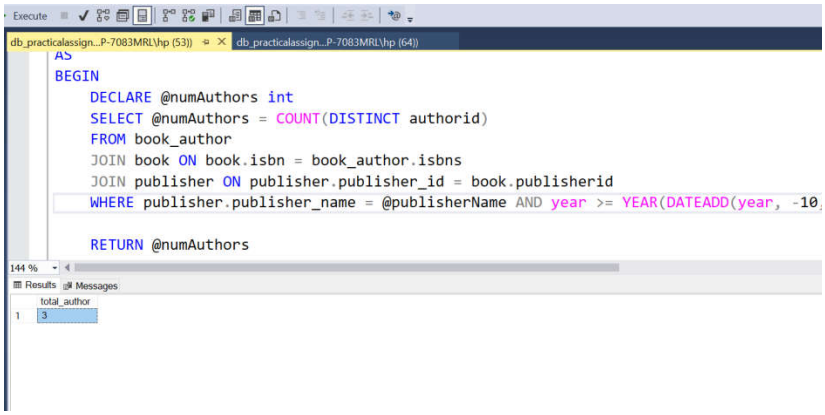
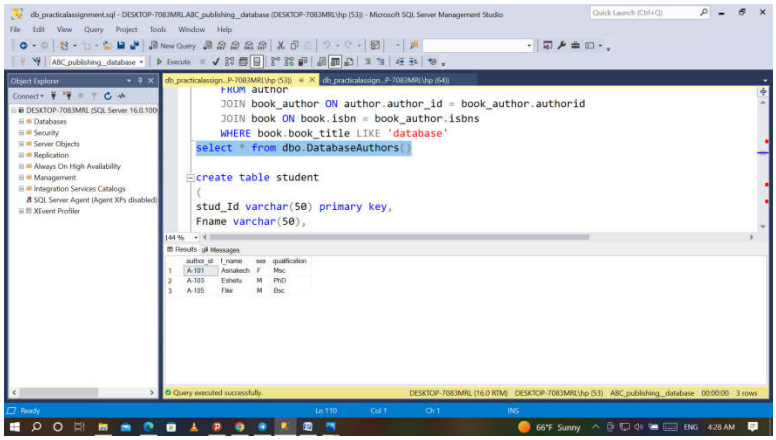
```sql
create database ABC_publishing__database
use ABC_publishing__database

create table book_author
(
isbns varchar(50),
authorid varchar(50),
year int,
foreign key(authorid)references author(author_id),
foreign key(isbns)references book(isbn)
)
create table book
(
isbn varchar(50) primary key,
book_title varchar(50),
unit_price money,
publisherid varchar(50),
foreign key(publisherid)references publisher(publisher_id)
)
create table author
(
author_id varchar(50) primary key,
f_name varchar(50),
sex char,
qualification varchar(50)
)
create table publisher
(
publisher_id varchar(50) primary key,
publisher_name varchar(50),
city varchar(50)
)
select publisher_id,qualification,year,f_name,sex,publisher_name,book_title from
book_author
join author on author.author_id=book_author.authorid
join book on book.isbn=book_author.isbns
join publisher on publisher.publisher_id=book.publisherid
where year>2014 and publisher_name
select * from book_author
insert into book_author values
('1-84356-028-4','A-102',2015),
 ('1-84356-028-2','A-101',2019),
('1-84356-028-7','A-103',2020),
('1-84356-028-8','A-100',2018),
('1-84356-028-3','A-105',2016),
('1-84356-028-5','A-104',2017),
('1-84356-028-1','A-106',2021),
('1-84356-028-10','A-108',2014)
select * from author
insert into author values
('A-100','Betelhem','F','PhD'),
('A-101','Asnakech','F','Msc'),
('A-102','Dereb','M','Msc'),
('A-103','Eshetu','M','PhD'),
('A-104','Haymanot','F','Bsc'),
('A-105','Fikir','M','Bsc'),
('A-106','Alex','M','Bsc'),
('A-107','Dawit','M','PhD'),
```

```
('A-108','Solomon','M','PhD'),
('A-109','Amenadab','M','PhD')
select * from publisher
insert into publisher values
('P-100','ABC printing press','Adiss ababa'),
('P-101','Adis printing','Nazret'),
('P-102','Adebabay Busines','Bahr dar'),
('P-103','Admas Advertising','Gonder'),
('P-104','Addis ababa university press','Mekele'),
('P-105','Bicolo printing ','Hawassa')
select * from book
insert into book values
('1-84356-028-1','Ethiopian history',250.00,'p-100'),
('1-84356-028-10','programing',630.00,'p-105'),
('1-84356-028-2','database',300.00,'p-103'),
('1-84356-028-3','database',450.00,'p-104'),
('1-84356-028-4','c++',390.00,'p-101'),
('1-84356-028-5','java',500.00,'p-100'),
('1-84356-028-6','IP',600.00,'p-105'),
('1-84356-028-7','database',1000.00,'p-100'),
('1-84356-028-8','SAD',720.00,'p-101'),
('1-84356-028-9','MIS',850.00,'p-100')

/*1.1 Write a scalar function that takes Publisher_Name as an argument and return total
number of authors who published books in the past ten years*/
CREATE FUNCTION getNumAuthors(@publisherName varchar(50))
RETURNS int
AS
BEGIN
    DECLARE @numAuthors int
    SELECT @numAuthors = COUNT(DISTINCT authorid)
    FROM book_author
    JOIN book ON book.isbn = book_author.isbns
    JOIN publisher ON publisher.publisher_id = book.publisherid
    WHERE publisher.publisher_name = @publisherName AND year >= YEAR(DATEADD(year, -10,
GETDATE()))

    RETURN @numAuthors
END
select dbo.getNumAuthors('abc printing press') as total_author
```



```
/*1.2 Create an inline table valued function that generates the list of all authors who
published
Database books.*/

CREATE FUNCTION DatabaseAuthors()
RETURNS TABLE
AS
RETURN
    SELECT DISTINCT author.*
    FROM author
    JOIN book_author ON author.author_id = book_author.authorid
    JOIN book ON book.isbn = book_author.isbns
    WHERE book.book_title LIKE 'database'
select * from dbo.DatabaseAuthors()
```



```
create table student
(
stud_Id varchar(50) primary key,
Fname varchar(50),
Lname varchar(50),
sex char,
pocket_maney money,
Dnum varchar(50),
CGPA float,
foreign key(Dnum)references department(Dno)
)
create table department
(
Dno varchar(50) primary key,
Dname varchar(50)
)
create table student_Grade
(
stud_id varchar(50),
cours_code varchar(50),
grade char,
foreign key(stud_id)references student(stud_Id),
foreign key(cours_code)references course(course_code)
)
```

```sql
create table course
(
course_code varchar(50) primary key,
course_title varchar(50),
cr_Hr int
)
insert into course values
('CS100','introduction to computer science',2),
('CS200','introduction to information techhnology',2),
('CS300','object oriented programing',3),
('CS310','advanced database system',4),
('CS400','fundamental of database system',3),
('EMT100','introduction to emerging technology',2),
('IT101','disributed system',4),
('IT202','multimedia',3),
('IT301','data communication',4),
('IT423','operating system',4),
('SE303','introduction to software engineering',3),
('SE401','software testing',4)


insert into department values
('Dept 100','computing'),
('Dept 101','civil engineering'),
('Dept 102','mechanical engineerung'),
('Dept 103','chemical enginnerig'),
('Dept 104','electrical engineering')

insert into student values
('ST101','Betelhem','Dawit','F',300.00,'Dept 100',4.00),
('ST102','Asnakech','Worku','F',200.00,'Dept 103',3.00),
('ST103','Dereb','Asmamaw','M',500.00,'Dept 104',3.50),
('ST104','Eshetu','Amare','M',600.00,'Dept 100',3.75),
('ST105','Girma','Moges','M',200.00,'Dept 102',2.00),
('ST106','Haymanot','Degu','F',300.00,'Dept 103',3.56),
('ST107','Alem','Tefera','F',400.00,'Dept 101',2.00),
('ST108','Zebidwr','Yitagesu','F',400.00,'Dept 100',2.78),
('ST109','Habtam','Tigabu','F',300.00,'Dept 101',3.32),
('ST110','Fikir','AShenafi','M',500.00,'Dept 104',3.70),
('ST111','Befkadu','Honelign','M',600.00,'Dept 102',3.00)


insert into student_Grade values
('ST101','CS200','A'),
('ST108','CS310','A'),
('ST104','SE303','B'),
('ST110','CS100','C'),
('ST111','EMT100','B'),
('ST103','CS310','B'),
('ST106','IT202','C'),
('ST109','IT101','B'),
('ST102','SE401','A'),
('ST105','CS300','B'),
('ST107','IT423','A'),
('ST101','IT301','C')

/*
2.1 Write a stored procedure that increment Pocket_Money by 50% for excellent students
```

```
whose CGPA is above 3.75. Use studentID as a parameter to the procedure.
*/
CREATE PROCEDURE sp_IncrementPocketMoney
    @studentID VARCHAR(50)
AS
BEGIN
    SET NOCOUNT ON;

    DECLARE @cgpa FLOAT;
    DECLARE @pocketMoney MONEY;

    -- Get the student's CGPA and pocket money
    SELECT @cgpa = CGPA, @pocketMoney = pocket_maney
    FROM student
    WHERE stud_Id = @studentID;

    -- Check if the student is excellent and has a CGPA above 3.75
    IF @cgpa > 3.75
    BEGIN
        -- Increment the pocket money by 50%
        SET @pocketMoney = @pocketMoney * 1.5;

        -- Update the student's pocket money
        UPDATE student
        SET pocket_maney = @pocketMoney
        WHERE stud_Id = @studentID;

        PRINT 'Pocket money incremented by 50% for student ' + @studentID;
    END
    ELSE
    BEGIN
        PRINT 'Student ' + @studentID + ' is not eligible for pocket money increment';
    END
END
--example for ST101
exec sp_IncrementPocketMoney'ST101'---Pocket money incremented by 50% for student ST101
--example for ST105
exec sp_IncrementPocketMoney'ST105'--Student ST105 is not eligible for pocket money
increment


/*
2.2 Create a trigger called Pocket_money_Limit that prevent records if students
Pocket money to be pay is greater than 600 and less than 200 Birr
*/
CREATE TRIGGER Pocket_money_Limit
ON student
AFTER INSERT, UPDATE
AS
BEGIN
    IF EXISTS (
        SELECT 1
        FROM inserted
        WHERE pocket_maney > 600 OR pocket_maney < 200
    )
    BEGIN
        RAISERROR('Pocket money should be between 200 and 600 Birr', 16, 1);
        ROLLBACK TRANSACTION;
```

```
        RETURN;
    END
END

insert into student
---here is a pocket money value outside the allowed range
-- This insert statement should fail due to the pocket money value being too high
INSERT INTO student (stud_Id, Fname, Lname, sex, pocket_maney, Dnum, CGPA)
VALUES ('ST112', 'Lidya', 'Tadesse', 'F', 700.00, 'Dept 100', 3.50);

-- This update statement should also fail due to the pocket money value being too low
UPDATE student
SET pocket_maney = 100.00
WHERE stud_Id = 'ST102';
```