From the above given tables perform the following queries:

Simple View

Part – A

1. Create a view Personal with all columns.
```
CREATE VIEW Personal
AS
SELECT * FROM Student;
```

2. Create a view Student_Details having columns Name, Branch & SPI.
```
CREATE VIEW Student_Details
AS
SELECT Name, Branch, SPI FROM Student;
```

3. Create a view Academic having columns RNo, Name, Branch.
```
CREATE VIEW Academic
AS
SELECT Rno, Name, Branch FROM Student;
```

4. Create a view Student_Data having all columns but students whose bklogs are more than 2.
```
CREATE VIEW Student_Data
AS
SELECT * FROM Student
WHERE Bklog <= 2;
```

5. Create a view Student_Pattern having RNo, Name & Branch columns in which Name consists of four letters.
```
CREATE VIEW Student_Pattern
AS
SELECT Rno, Name, Branch FROM Student
WHERE LEN(Name) = 4;
```

Part – B

6. Insert a new record to Academic view. (107, Meet, ME). Remaining all columns must be null.
```
INSERT INTO Academic (Rno, Name, Branch)
VALUES (107, 'Meet', 'ME');
```

7. Update the branch of Amit from CE to ME in Student_Details view
```
UPDATE Student_Details
SET Branch = 'ME'
WHERE Name = 'Amit';
```

8. Delete a student whose roll number is 104 from Academic view.
```
DELETE FROM Academic
```

```
WHERE Rno = 104;
```

9. Create a view that displays information of all students whose SPI is above 8.5.

```
CREATE VIEW High_SPI_Students
AS
SELECT * FROM Student
WHERE SPI > 8.5;
```

10. Create a view that displays 0 backlog students.

```
CREATE VIEW Zero_Backlog_Students
AS
SELECT * FROM Student
WHERE Bklog = 0;
```

Part – C

11. Create a view Computer that displays CE branch data only.

```
CREATE VIEW Computer
AS
SELECT * FROM Student
WHERE Branch = 'CE';
```

12. Create a view Result_EC that displays the name and SPI of students with SPI less than 5 of branch EC.

```
CREATE VIEW Result_EC
AS
SELECT Name, SPI FROM Student
WHERE SPI < 5 AND Branch = 'EC';
```

13. Update the result of student Sanjay to 4.90 in Result_EC view.

```
UPDATE Result_EC
SET SPI = 4.90
WHERE Name = 'Sanjay';
```

14. Create a view Stu_Bklog with RNo, Name and Bklog columns in which name starts with 'M' and having bklogs more than 5.

```
CREATE VIEW Stu_Bklog
AS
SELECT Rno, Name, Bklog FROM Student
WHERE Name LIKE 'M%' AND Bklog > 5;
```

15. Drop Computer view from the database.

```
DROP VIEW Computer;
```

Complex View

Part – A

1. Create view that displays all the customers along with their corresponding account balances.

```
CREATE VIEW CustomerAccounts
AS
SELECT c.CustomerID, c.FirstName, c.LastName, a.Balance
FROM Customer c LEFT JOIN Account a
ON c.CustomerID = a.CustomerID;
```

2. Create view that displays total balance for each customer.

```
CREATE VIEW TotalBalancePerCustomer
AS
SELECT c.CustomerID, c.FirstName,  c.LastName, SUM(a.Balance) AS TotalBalance
FROM Customer c LEFT JOIN Account a
ON c.CustomerID = a.CustomerID
GROUP BY c.CustomerID, c.FirstName, c.LastName;
```

3. Create view that displays customers who have multiple accounts.

```
CREATE VIEW CustomersWithMultipleAccounts
AS
SELECT c.CustomerID, c.FirstName, c.LastName, COUNT(a.AccountID) AS AccountCount
FROM Customer c INNER JOIN Account a
ON c.CustomerID = a.CustomerID
GROUP BY c.CustomerID, c.FirstName, c.LastName
HAVING COUNT(a.AccountID) > 1;
```

Part – B

4. Create a view that displays customer details who have an account created in the last month.

```
CREATE VIEW RecentAccountHolders
AS
SELECT c.CustomerID, c.FirstName, c.LastName, a.AccountID, a.CreatedDate
FROM Customer c INNER JOIN Account a
ON c.CustomerID = a.CustomerID
WHERE a.CreatedDate >= DATEADD(MONTH, -1, GETDATE());
```

5. Create a view that displays customers who have the highest account balance.

```
CREATE VIEW HighestAccountBalance
AS
SELECT c.CustomerID, c.FirstName, c.LastName, a.Balance AS HighestBalance
FROM Customer c INNER JOIN Account a
ON c.CustomerID = a.CustomerID
WHERE a.Balance = (SELECT MAX(Balance) FROM Account);
```

Part – C

6. Create a view that displays name of the customers whose account balance is between 5000 to 10000 and account type is Saving.

```
CREATE VIEW CustomersWithSavingAccounts
AS
SELECT c.CustomerID, c.FirstName, c.LastName, a.Balance
FROM Customer c INNER JOIN Account a
ON c.CustomerID = a.CustomerID
WHERE a.Balance BETWEEN 5000 AND 10000 AND a.AccountType = 'Saving';
```

7. Create a view that displays minimum and maximum balance for each customer.

```
CREATE VIEW MinMaxBalancePerCustomer
AS
SELECT c.CustomerID, c.FirstName, c.LastName,
        MIN(a.Balance) AS MinBalance,
        MAX(a.Balance) AS MaxBalance
FROM Customer c LEFT JOIN Account a
ON c.CustomerID = a.CustomerID;
```