1. Users Table

```
CREATE TABLE Users (
    Name VARCHAR(50),
    Email_ID VARCHAR(50),
    Username VARCHAR(30),
    Password VARCHAR(30),
    User_Id INT PRIMARY KEY,
    User_type VARCHAR(20),
    Reg_date DATE,
    Last_login DATE);


INSERT INTO Users VALUES
('Amit', 'amit@mail.com', 'amit01', 'pass123',1,’Student’,
TO_DATE('2025-01-10','YYYY-MM-DD'),TO_DATE('2025-08-01', 'YYYY-MM-DD'));
('Bavani', 'bava@mail.com', 'bava88', 'bavpass', 2, 'Teacher',
 TO_DATE('2025-02-12', 'YYYY-MM-DD'),TO_DATE('2025-08-02', 'YYYY-MM-DD'));
('Charlie', 'charlie@mail.com', 'charlie76', 'char123', 3, 'Student',
 TO_DATE('2025-03-15', 'YYYY-MM-DD'),TO_DATE('2025-08-03', 'YYYY-MM-DD'));
('David', 'david@mail.com', 'david44', 'davpass', 4, 'Admin',
 TO_DATE('2025-01-25', 'YYYY-MM-DD'),TO_DATE('2025-08-04', 'YYYY-MM-DD'));
('Emma', 'emma@mail.com', 'ema99', 'emapass', 5, 'Student',
 TO_DATE('2025-04-01', 'YYYY-MM-DD'),TO_DATE('2025-08-05', 'YYYY-MM-DD'));
('Fathima', 'fathima@mail.com', 'fathima78', 'fathupass', 6, 'Student',
 TO_DATE('2025-05-12', 'YYYY-MM-DD'), TO_DATE('2025-08-06', 'YYYY-MM-DD'));
```

2. Course Table

```
CREATE TABLE Course (
   User_Id INT,
   Course_Id INT PRIMARY KEY,
   Course_name VARCHAR(50),
   Description VARCHAR2(500),
   Enrollment_date DATE);


INSERT INTO Course VALUES
(1, 101, 'Python Basics', 'Intro to Python', TO_DATE('2025-01-15', 'YYYY-MM-DD')),
(2, 102, 'Web Development', 'HTML, CSS, JS basics', TO_DATE('2025-02-20', 'YYYY-MM-DD')),
(3, 103, 'Database Fundamentals', 'SQL Basics', TO_DATE('2025-03-22', 'YYYY-MM-DD')),
(4, 104, 'Java Programming', 'Core Java concepts', TO_DATE('2025-01-25', 'YYYY-MM-DD')),
(5, 105, 'UI/UX Design', 'Design fundamentals', TO_DATE('2025-04-05', 'YYYY-MM-DD')),
(6, 106, 'Machine Learning', 'Intro to ML concepts', TO_DATE('2025-05-15', 'YYYY-MM-DD'));
```

3. Progress Table

```
CREATE TABLE Progress (
   User_Id INT,
   Enrollment_date DATE,
   Course_Id INT,
   Status VARCHAR(20));


INSERT INTO Progress VALUES
(1, TO_DATE('2025-01-15', 'YYYY-MM-DD'), 101, 'Completed'),
(2, TO_DATE('2025-02-20', 'YYYY-MM-DD'), 102, 'Ongoing'),
(3, TO_DATE('2025-03-22', 'YYYY-MM-DD'), 103, 'Ongoing'),
(4, TO_DATE('2025-01-25', 'YYYY-MM-DD'), 104, 'Completed'),
(5, TO_DATE('2025-04-05', 'YYYY-MM-DD'), 105, 'Not Started'),
(6, TO_DATE('2025-05-15', 'YYYY-MM-DD'), 106, 'Ongoing');
```

4. Assignment Table

```
CREATE TABLE Assignment (
   User_Id INT,
   Course_Id INT,
   Assignment_Id INT PRIMARY KEY,
   Assignment_type VARCHAR(20),
   Description VARCHAR2(500),
   Total_marks INT);


INSERT INTO Assignment VALUES
(1, 101, 201, 'Quiz', 'Python Basics Quiz', 20),
(2, 102, 202, 'Project', 'Create a portfolio website', 50),
(3, 103, 203, 'Quiz', 'SQL Queries Test', 20),
(4, 104, 204, 'Assignment', 'Java OOP concepts', 30),
(5, 105, 205, 'Project', 'App design prototype', 40),
(6, 106, 206, 'Quiz', 'Intro ML concepts', 25);
```

5. Submission Table

```
CREATE TABLE Submission (
   User_Id INT,
   Assignment_Id INT,
   Submission_Id INT PRIMARY KEY,
   Submitted_on DATE,
   Score INT,
   Feedback VARCHAR2(500);


INSERT INTO Submission VALUES
(1, 201, 301, TO_DATE('2025-01-20', 'YYYY-MM-DD'), 18, 'Good job'),
(2, 202, 302, TO_DATE('2025-02-28', 'YYYY-MM-DD'), 45, 'Well done'),
(3, 203, 303, TO_DATE('2025-03-25', 'YYYY-MM-DD'), 15, 'Needs improvement'),
(4, 204, 304, TO_DATE('2025-01-30', 'YYYY-MM-DD'), 28, 'Excellent'),
(5, 205, 305, TO_DATE('2025-04-15', 'YYYY-MM-DD'), 35, 'Creative work'),
(6, 206, 306, TO_DATE('2025-05-20', 'YYYY-MM-DD'), 20, 'Nice effort');
```

6. Course_Content Table

```
CREATE TABLE Course_Content (
   User_Id INT,
   Course_Id INT,
   Content_Id INT PRIMARY KEY,
   Content_type VARCHAR(20),
   Title VARCHAR(50),
   Module_Id INT,
   Module_title VARCHAR(50));


INSERT INTO Course_Content VALUES
(1, 101, 401, 'Video', 'Intro to Python', 501, 'Getting Started'),
(2, 102, 402, 'PDF', 'HTML Basics', 502, 'Web Structure'),
(3, 103, 403, 'Video', 'SQL Basics', 503, 'Database Intro'),
(4, 104, 404, 'PDF', 'Java Basics', 504, 'Java Intro'),
(5, 105, 405, 'Video', 'UI Principles', 505, 'Design Basics'),
(6, 106, 406, 'PDF', 'ML Concepts', 506, 'Machine Learning Intro');
```

DATA DEFINITION LANGUAGE

```
SQL> ALTER TABLE Users ADD Phone_Number VARCHAR(15);

Table altered.


SQL> SELECT * FROM Users;

NAME       EMAIL_ID                         USERNAME        PASSWORD        USER_ID
---------- -------------------------------- --------------- --------------- ----------
Amit       amit@mail.com                    amit01          pass123         1
```

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	10-JAN-25	01-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Bavani	bava@mail.com	bava88	bavpass	2

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Teacher	12-FEB-25	02-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Charlie	charlie@mail.com	charlie76	char123	3

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	15-MAR-25	03-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
David	david@mail.com	david44	davpass	4

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Admin	25-JAN-25	04-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Emma	emma@mail.com	ema99	emapass	5

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	01-APR-25	05-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Fathima	fathima@mail.com	fathima78	fathupass	6

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	12-MAY-25	06-AUG-25	

6 rows selected.

SQL> DROP TABLE Users;

Table dropped.

SQL> TRUNCATE TABLE Users;

Table truncated.

DATA MANIPULATION LANGUAGE

SQL> UPDATE Users SET User_type = 'Teacher' WHERE User_Id = 1;

1 row updated.

SQL> SELECT*FROM Users;

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Amit	amit@mail.com	amit01	pass123	1

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	10-JAN-25	01-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Bavani	bava@mail.com	bava88	bavpass	2

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Teacher	12-FEB-25	02-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Charlie	charlie@mail.com	charlie76	char123	3

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	15-MAR-25	03-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
David	david@mail.com	david44	davpass	4

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Admin	25-JAN-25	04-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Emma	emma@mail.com	ema99	emapass	5

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	01-APR-25	05-AUG-25	

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Fathima	fathima@mail.com	fathima78	fathupass	6

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	12-MAY-25	06-AUG-25	

6 rows selected.

```
SQL> DELETE FROM Users WHERE User_Id = 6;

1 row deleted.


SQL> SELECT*FROM Users;

NAME       EMAIL_ID                  USERNAME        PASSWORD        USER_ID
---------- ------------------------- --------------- --------------- ----------
Amit       amit@mail.com             amit01          pass123         1

USER_TYPE  REG_DATE  LAST_LOGIN  PHONE_NUMBER
---------- --------- ----------- ------------
Student    10-JAN-25 01-AUG-25

NAME       EMAIL_ID                  USERNAME        PASSWORD        USER_ID
---------- ------------------------- --------------- --------------- ----------
Bavani     bava@mail.com             bava88          bavpass         2

USER_TYPE  REG_DATE  LAST_LOGIN  PHONE_NUMBER
---------- --------- ----------- ------------
Teacher    12-FEB-25 02-AUG-25

NAME       EMAIL_ID                  USERNAME        PASSWORD        USER_ID
---------- ------------------------- --------------- --------------- ----------
Charlie    charlie@mail.com          charlie76       char123         3

USER_TYPE  REG_DATE  LAST_LOGIN  PHONE_NUMBER
---------- --------- ----------- ------------
Student    15-MAR-25 03-AUG-25

NAME       EMAIL_ID                  USERNAME        PASSWORD        USER_ID
---------- ------------------------- --------------- --------------- ----------
David      david@mail.com            david44         davpass         4

USER_TYPE  REG_DATE  LAST_LOGIN  PHONE_NUMBER
---------- --------- ----------- ------------
Admin      25-JAN-25 04-AUG-25
```

NAME	EMAIL_ID	USERNAME	PASSWORD	USER_ID
Emma	emma@mail.com	ema99	emapass	5

USER_TYPE	REG_DATE	LAST_LOGIN	PHONE_NUMBER
Student	01-APR-25	05-AUG-25	

SQL> ALTER TABLE Course ADD Duration_Weeks INT;

Table altered.

SQL> SELECT*FROM COURSE;

USER_ID	COURSE_ID	COURSE_NAME	DESCRIPTION	ENROLLMENT
1	101	Python Basics	Intro to Python	15-JAN-25

USER_ID	COURSE_ID	COURSE_NAME	DESCRIPTION	ENROLLMENT
2	102	Web Development	HTML, CSS, JS basics	20-FEB-25

USER_ID	COURSE_ID	COURSE_NAME	DESCRIPTION	ENROLLMENT
3	103	Database Fundamentals	SQL Basics	22-MAR-25

USER_ID	COURSE_ID	COURSE_NAME	DESCRIPTION	ENROLLMENT
4	104	Java Programming	Core Java concepts	25-JAN-25

USER_ID	COURSE_ID	COURSE_NAME	DESCRIPTION	ENROLLMENT
5	105	UI/UX Design	Design fundamentals	05-APR-25

USER_ID	COURSE_ID	COURSE_NAME	DESCRIPTION	ENROLLMENT
6	106	Machine Learning	Intro to ML concepts	15-MAY-25

DATA INTEGRITY CONSTRAINTS

```
SQL> ALTER TABLE Users ADD CONSTRAINT PK_Users PRIMARY KEY (User_Id);

Table altered.

SQL> ALTER TABLE Course ADD CONSTRAINT UQ_CourseName UNIQUE (Course_name);

Table altered.

SQL> ALTER TABLE Course_Content ADD CONSTRAINT FK_Content_Course FOREIGN
KEY (Course_Id) REFERENCES Course(Course_Id);

Table altered.

SQL> ALTER TABLE Assignment ADD CONSTRAINT CHK_TotalMarks CHECK
(Total_marks >= 0);

Table altered.

SQL> ALTER TABLE Users ALTER COLUMN User_type SET DEFAULT 'Student';

Table altered.

SQL> ALTER TABLE Submission ALTER COLUMN Submission_Id SET NOT NULL;

Table altered.
```

AGGREGATE FUNCTIONS AND SORTING

```
SQL> SELECT COUNT(*) AS Total_Users FROM Users;

TOTAL_USERS
-----------
          6


SQL> SELECT User_type, COUNT(*) AS Count FROM Users GROUP BY User_type;

USER_TYPE    COUNT
-------------------  -----------
Student                5
Teacher                1


SQL> SELECT MIN(Reg_date) AS Earliest_Registration,MAX(Reg_date) AS
Latest_Registration FROM Users;

EARLIEST_REGISTRATION      LATEST_REGISTRATION
---------------------------------------    ———————————————————————————
            10-JAN-25                        12-MAY-25


SQL> SELECT COUNT(*) AS Total_Enrollments FROM Course;

TOTAL_ENROLLMENTS
---------------------------------
               6


SQL> SELECT Course_Name, COUNT(User_Id) AS Enrolled_Users FROM Course GROUP
BY Course_Name;

COURSE_NAME       ENROLLED_USERS
----------------------       ----------------------------
Web Development                 1
Database Fundamentals           1
Java Programming                1
UI/UX Design                    1
Machine Learning                1
```

6 rows selected.

```
SQL> SELECT MIN(Enrollment_Date) AS Earliest,MAX(Enrollment_Date) AS Latest FROM
Course;

EARLIEST      LATEST
--------------    --------------
15-JAN-25     15-MAY-25


SQL> SELECT COUNT(*) AS Total_Progress_Records FROM Progress;

TOTAL_PROGRESS_RECORDS
----------------------------------------------
                  6


SQL> SELECT Status, COUNT(*) AS User_Count FROM Progress GROUP BY Status;

STATUS       USER_COUNT
-----------      ---------------------
Completed            2
Ongoing              3
Not Started          1


SQL> SELECT Course_Id, COUNT(User_Id) AS Enrolled_Users FROM Progress GROUP BY
Course_Id;

 COURSE_ID      ENROLLED_USERS
------------------     ——————————————————
      101                    1
      102                    1
      103                    1
      104                    1
      105                    1
      106                    1

6 rows selected.
```

```
SQL> SELECT Course_Id, COUNT(*) AS Completed_Count FROM Progress WHERE Status =
'Completed' GROUP BY Course_Id;

 COURSE_ID      COMPLETED_COUNT
------------------    ——————————————
       101                    1
       104                    1


SQL> SELECT MIN(Enrollment_Date) AS Earliest_Enrollment,MAX(Enrollment_Date) AS
Latest_Enrollment FROM Progress;

EARLIEST_ENROLLMENT        LATEST_ENROLLMENT
-----------------------------------        --------------------------------
         15-JAN-25                          15-MAY-25


SQL> SELECT Status, COUNT(*) AS Count FROM Progress WHERE Course_Id = 101GROUP
BY Status;

 STATUS          COUNT
---------------      ——————————
Completed          1


SQL> SELECT COUNT(*) AS Total_Assignments FROM Assignment;

TOTAL_ASSIGNMENTS
---------------------------------
              6


SQL> SELECT User_Id, COUNT(*) AS Total_Assignment  FROM Assignment GROUP BY
User_Id;

   USER_ID          TOTAL_ASSIGNMENTS
---------------         --------------------------------
       1                        1
       2                        1
       3                        1
       4                        1
       5                        1
       6                        1
```

6 rows selected.

SQL> SELECT Course_Id,SUM(Total_Marks) AS Total_Marks, AVG(Total_Marks) AS Average_Marks FROM Assignment GROUP BY Course_Id;

COURSE_ID	TOTAL_MARKS	AVERAGE_MARKS
101	20	20
102	50	50
103	20	20
104	30	30
105	40	40
106	25	25

6 rows selected.

SQL> SELECT User_Id,SUM(Total_Marks) AS Total_Marks_Assigned FROM Assignment GROUP BY User_Id;

USER_ID	TOTAL_MARKS_ASSIGNED
1	20
2	50
3	20
4	30
5	40
6	25

6 rows selected.

SQL> SELECT User_Id,SUM(Total_Marks) AS Total_Marks_Assigned FROM Assignment GROUP BY User_Id;

USER_ID	TOTAL_MARKS_ASSIGNED
1	20
2	50
3	20
4	30
5	40
6	25

6 rows selected.

```
SQL> SELECT COUNT(*) AS Total_Submissions FROM Submission;

TOTAL_SUBMISSIONS
---------------------------------
              6


SQL> SELECT AVG(Score) AS Average_Score FROM Submission;

AVERAGE_SCORE
----------------------------
  26.8333333


SQL> SELECT MAX(Score) AS Highest_Score,MIN(Score) AS Lowest_Score FROM
Submission;

HIGHEST_SCORE     LOWEST_SCORE
-------------------------     -------------------------
         45                        15


SQL> SELECT User_Id,COUNT(*) AS Total_Submissions,SUM(Score) AS
Total_Score,AVG(Score) AS Average_Score FROM Submission GROUP BY User_Id


  USER_ID   TOTAL_SUBMISSIONS   TOTAL_SCORE   AVERAGE_SCORE
 ----------   -------------------------------   ---------------------   ----------------------------
       1                1                18                18
       2                1                45                45
       3                1                15                15
       4                1                28                28
       5                1                35                35
       6                1                20                20

6 rows selected.
```

```
SQL> SELECT Assignment_Id,COUNT(*) AS Total_Attempts,AVG(Score) AS Average_Score
FROM Submission GROUP BY Assignment_Id;

ASSIGNMENT_ID  TOTAL_ATTEMPTS   AVERAGE_SCORE
-------------  --------------  -------------
          201               1             18
          202               1             45
          203               1             15
          204               1             28
          205               1             35
          206               1             20

6 rows selected.

SQL> SELECT User_Id,SUM(Score) AS Total_Score FROM Submission GROUP BY User_Id
HAVING SUM(Score) > 40;

   USER_ID  TOTAL_SCORE
 ----------  -----------
         2           45


SQL> SELECT Submitted_On,COUNT(*) AS Submissions_On_Date FROM Submission
GROUP BY Submitted_On ORDER BY Submitted_On;

SUBMITTED       SUBMISSIONS_ON_DATE
---------       -------------------
20-JAN-25                         1
30-JAN-25                         1
28-FEB-25                         1
25-MAR-25                         1
15-APR-25                         1
20-MAY-25                         1

6 rows selected.
```

JOINS

```
SQL> SELECT Course.course_Id, Course.Course_name, Assignment.Assignment_Id,
Assignment.Assignment_type FROM Course INNER JOIN Assignment ON
Course.Course_Id Assignment.Course_Id;
```

COURSE_ID	COURSE_NAME	ASSIGNMENT_ID	ASSIGNMENT_TYPE
101	Python Basics	201	Quiz
102	Web Development	202	Project
103	Database Fundamentals	203	Quiz
104	Java Programming	204	Assignment
105	UI/UX Design	205	Project
106	Machine Learning	206	Quiz

6 rows selected.

```
SQL> SELECT Course.Course_Id, Course.Course_Name, Course.Description,
Course_Content.Content_type,Course_Content.Title FROM Course LEFT JOIN
Course_Content ON Course.Course_Id = Course_Content.Course_Id;
```

COURSE_ID	COURSE_NAME	DESCRIPTION	CONTENT_TYPE	TITLE
101	Python Basics	Intro to Python	Video	Intro to Python
102	Web Development	HTML, CSS, JS basics	PDF	HTML Basics
103	Database Fundamentals	SQL Basics	Video	SQL Basics
104	Java Programming	Java concepts	PDF	Java Basics
105	UI/UX Design	Design fundamentals	Video	UIPrinciples
106	Machine Learning	Intro to ML concepts	PDF	ML Concepts

6 rows selected.

SQL> SELECT Progress.User_Id, progress.Course_Id, Progress.Status,Users.Name, Users.Email_ID FROM Users RIGHT JOIN Progress ON Users.User_Id = Progress.User_Id;

USER_ID	COURSE_ID	STATUS	NAME	EMAIL_ID
1	101	Completed	Amit	amit@mail.com
2	102	Ongoing	Bavani	bava@mail.com
3	103	Ongoing	Charlie	charlie@mail.com
4	104	Completed	David	david@mail.com
5	105	Not Started	Emma	emma@mail.com
6	106	Ongoing	Fathima	fathima@mail.com

6 rows selected.

SQL> SELECT Users.User_Id, Users.Name, Users.Email_ID, Progress.Course_Id, Progress.Status FROM Users FULL OUTER JOIN Progress ON Users.User_Id = Progress.User_Id;

USER_ID	COURSE_ID	STATUS	NAME	EMAIL_ID
1	101	Completed	Amit	amit@mail.com
2	102	Ongoing	Bavani	bava@mail.com
3	103	Ongoing	Charlie	charlie@mail.com
4	104	Completed	David	david@mail.com
5	105	Not Started	Emma	emma@mail.com
6	106	Ongoing	Fathima	fathima@mail.com

6 rows selected.

SQL> SELECT Course.Course_Id, Course.Course_Name,Course_Content.title,
Course_Content.Module_title FROM Course CROSS JOIN Course_content;

COURSE_ID	COURSE_NAME	TITLE	MODULE_TITLE
101	Python Basics	Intro to Python	Getting Started
102	Web Development	Intro to Python	Getting Started
103	Database Fundamentals	Intro to Python	Getting Started
104	Java Programming	Intro to Python	Getting Started
105	UI/UX Design	Intro to Python	Getting Started
106	Machine Learning	Intro to Python	Getting Started

COURSE_ID	COURSE_NAME	TITLE	MODULE_TITLE
101	Python Basics	Intro to HTML Basics	Web Structure
102	Web Development	Intro to HTML Basics	Web Structure
103	Database Fundamentals	Intro to HTML Basics	Web Structure
104	Java Programming	Intro to HTML Basics	Web Structure
105	UI/UX Design	Intro to HTML Basics	Web Structure
106	Machine Learning	Intro to HTML Basics	Web Structure

COURSE_ID	COURSE_NAME	TITLE	MODULE_TITLE
101	Python Basics	SQL Basics	Database Intro
102	Web Development	SQL Basics	Database Intro
103	Database Fundamentals	SQL Basics	Database Intro
104	Java Programming	SQL Basics	Database Intro
105	UI/UX Design	SQL Basics	Database Intro
106	Machine Learning	SQL Basics	Database Intro

COURSE_ID	COURSE_NAME	TITLE	MODULE_TITLE
101	Python Basics	Java Basics	Java Intro
102	Web Development	Java Basics	Java Intro
103	Database Fundamentals	Java Basics	Java Intro
104	Java Programming	Java Basics	Java Intro
105	UI/UX Design	Java Basics	Java Intro
106	Machine Learning	Java Basics	Java Intro

COURSE_ID	COURSE_NAME	TITLE	MODULE_TITLE
101	Python Basics	UI Principles	Design Basics
102	Web Development	UI Principles	Design Basics
103	Database Fundamentals	UI Principles	Design Basics
104	Java Programming	UI Principles	Design Basics
105	UI/UX Design	UI Principles	Design Basics
106	Machine Learning	UI Principles	Design Basics

COURSE_ID	COURSE_NAME	TITLE	MODULE_TITLE
101	Python Basics	ML Concepts	Machine Learning Intro
102	Web Development	ML Concepts	Machine Learning Intro
103	Database Fundamentals	ML Concepts	Machine Learning Intro
104	Java Programming	ML Concepts	Machine Learning Intro
105	UI/UX Design	ML Concepts	Machine Learning Intro
106	Machine Learning	ML Concepts	Machine Learning Intro

36 rows selected.

SET OPERATIONS

SELECT * FROM Submission WHERE Score < 30;

USER_ID	ASSIGNMENT_ID	SUBMISSION_ID	SUBMITTED_ON	SCORE	FEEDBACK
1	201	301	20-JAN-25	18	Good job
3	203	303	25-MAR-25	15	Needs Improvement
4	204	304	30-JAN-25	28	Excellent
6	206	306	20-MAY-25	20	Nice effort

SQL> SELECT * FROM Submission WHERE Score <= 20;

USER_ID	ASSIGNMENT_ID	SUBMISSION_ID	SUBMITTED_ON	SCORE	FEEDBACK
1	201	301	20-JAN-25	18	Good job
3	203	303	25-MAR-25	15	Needs Improvement
6	206	306	20-MAY-25	20	Nice effort

SQL> SELECT User_Id, Submitted_on, Score FROM Submission WHERE Score BETWEEN 20 AND 30;

USER_ID	SUBMITTED_ON	SCORE
4	30-JAN-25	28
6	20-MAY-25	20

```
SQL> SELECT User_Id, Submitted_on, Score FROM Submission WHERE Score IN (18, 20,
28);

   USER_ID  SUBMITTED_ON  SCORE
 ---------- ——————————— ——————
         1             20-JAN-25          18
         4             30-JAN-25          28
         6             20-MAY-25          20


SQL> SELECT User_Id, Submitted_on, Score FROM Submission WHERE Score IN (18, 20,
28);

   USER_ID SUBMITTED_ON  SCORE
---------- ------------------------ ------------
         1             20-JAN-25          18
         4             30-JAN-25          28
         6             20-MAY-25          20


SQL> SELECT User_Id, Score FROM Submission WHERE User_Id = 1 AND Score = 20;

no rows selected

SQL> SELECT User_Id, Score FROM Submission WHERE User_Id = 4 AND Score = 28;

   USER_ID  SCORE
---------- ——————
         4             28


SQL> SELECT User_Id, Score, (Score - 5) AS Adjusted_Score FROM Submission;

   USER_ID SCORE ADJUSTED_SCORE
---------- ---------- ------------------------------
         1          18                     13
         2          45                     40
         3          15                     10
         4          28                     23
         5          35                     30
         6          20                     15

6 rows selected.
```

```
SQL> SELECT User_Id, TO_CHAR(Score) AS Info  FROM Submission UNION SELECT
User_Id, 'Submitted' AS Info  FROM Submission;

   USER_ID       INFO
---------- ----------
         1         18
         2         45
         3         15
         4         28
         5         35
         6         20
         1  Submitted
         2  Submitted
         3  Submitted
         4  Submitted
         5  Submitted
         6  Submitted

12 rows selected.


SQL> SELECT User_Id FROM Submission INTERSECT SELECT User_Id FROM
Course_Content;

   USER_ID
----------
         1
         2
         3
         4
         5
         6

6 rows selected.
```

```
SQL> SELECT User_Id, Score FROM Submission WHERE Score > (SELECT AVG(Score)
FROM Submission);

   USER_ID SCORE
---------- ----------
         2         45
         4         28
         5         35


SQL> SELECT User_Id, Submitted_on FROM Submission WHERE Submitted_on = (SELECT
MIN(Submitted_on) FROM Submission);

   USER_ID SUBMITTED_ON
---------- ---------
         1 20-JAN-25


SQL> SELECT User_Id, Assignment_Id FROM Submission WHERE Assignment_Id = 205;

   USER_ID ASSIGNMENT_ID
---------- -------------
         5           205


SQL> SELECT User_Id, Score FROM Submission WHERE Score BETWEEN 20 AND 35;

 USER_ID SCORE
---------- ----------
         4         28
         5         35
         6         20


SQL> SELECT User_Id, Submitted_on FROM Submission WHERE Submitted_on = (SELECT
MAX(Submitted_on) FROM Submission);

   USER_ID SUBMITTED_ON
---------- ---------
         6 20-MAY-25
```

```
SQL> SELECT User_Id, Assignment_Id  FROM Submission WHERE Assignment_Id = 206;

   USER_ID  ASSIGNMENT_ID
 ---------- ------------------------
         6                206
```

```
SQL> SELECT User_Id, Score FROM Submission WHERE Score < 20 OR Score > 40;

   USER_ID   SCORE
 ----------  ----------
         1          18
         2          45
         3          15
```

```
SQL> SELECT User_Id, Score FROM Submission WHERE Score > 40;

   USER_ID    SCORE
 ----------  -----------
         2           45
```

VIEWS

```
SQL> CREATE VIEW UserCourseStatus AS SELECT User_Id, Course_Id, Status FROM
Progress;

View created.

SQL> CREATE VIEW HighMarkAssignments AS SELECT User_Id, Course_Id, Assignment_Id,
Total_marks FROM Assignment  WHERE Total_marks > 30;

View created.

SQL> CREATE VIEW UserAssignmentTypes AS SELECT User_Id, Assignment_Id,
Assignment_type FROM Assignment;

View created.
```

```
SQL> CREATE VIEW OngoingCourses AS SELECT User_Id, Course_Id FROM Progress
WHERE Status = 'Ongoing';

View created.

SQL> CREATE VIEW UserSubmissionCounts AS SELECT User_Id, COUNT(Submission_Id)
AS Total_Submissions FROM Submission GROUP BY User_Id;

View created.

SQL> CREATE VIEW QuizAssignments AS SELECT Assignment_Id, User_Id, Course_Id
FROM Assignment WHERE Assignment_type = 'Quiz';

View created.

SQL> CREATE VIEW HighScoreSubmissions AS SELECT Submission_Id, User_Id, Score
FROM Submission WHERE Score > 25;

View created.

SQL> CREATE VIEW NotStartedUsers AS SELECT User_Id, Course_Id FROM Progress
WHERE Status = 'Not Started';

View created.

SQL> CREATE VIEW AssignmentDescriptions AS SELECT User_Id, Assignment_Id,
Description FROM Assignment;

View created.

 SQL> CREATE VIEW Course102Users AS SELECT User_Id FROM Progress WHERE
Course_Id = 102;

View created.
```