1. Create Table Query with 10 Students Records

```
CREATE TABLE Students (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(50),
    Age INT,
    Gender VARCHAR(10),
    Grade VARCHAR(2),
    Marks INT
);
```

This query creates a `Students` table with columns:

- `StudentID`: Unique identifier for each student.
- `Name`: Name of the student.
- `Age`: Age of the student.
- `Gender`: Gender of the student.
- `Grade`: Grade the student achieved.
- `Marks`: Marks the student received in an exam.

2. Insert Data into Table

```
INSERT INTO Students (StudentID, Name, Age, Gender, Grade, Marks)
VALUES (1, 'John Doe', 20, 'Male', 'A', 90),
       (2, 'Jane Smith', 22, 'Female', 'B', 75),
       (3, 'Mark Johnson', 21, 'Male', 'A', 85),
       (4, 'Emily Davis', 23, 'Female', 'C', 60),
       (5, 'Michael Brown', 20, 'Male', 'B', 80),
       (6, 'Sarah Wilson', 22, 'Female', 'A', 92),
       (7, 'David Moore', 21, 'Male', 'C', 65),
       (8, 'Olivia Taylor', 20, 'Female', 'B', 78),
       (9, 'James Lee', 22, 'Male', 'A', 88),
       (10, 'Isabella Clark', 23, 'Female', 'B', 82);
```

This query inserts 10 records into the `Students` table with the following columns: `StudentID`, `Name`, `Age`, `Gender`, `Grade`, and `Marks`.

3. Update Query

```
UPDATE Students
SET Marks = 95
WHERE StudentID = 1;
```

This query updates the `Marks` of the student with `StudentID` 1 to 95. It shows how to modify an existing record based on a specific condition (in this case, where `StudentID = 1`).

4. Alter Table Query (Adding a New Column)

```
ALTER TABLE Students
ADD Email VARCHAR(100);
```

This query alters the `Students` table by adding a new column `Email` with a data type of `VARCHAR(100)`. It's used to add new attributes to an existing table.

5. WHERE Clause with "AND"

```
SELECT * FROM Students
WHERE Age > 21 AND Grade = 'B';
```

This query retrieves records where the age of the student is greater than 21 *AND* their grade is 'B'. The `AND` operator is used to apply multiple conditions together.

6. WHERE Clause with "OR"

```
SELECT * FROM Students
WHERE Grade = 'A' OR Marks > 80;
```

This query retrieves records where the student has a grade 'A' *OR* their marks are greater than 80. The `OR` operator is used to combine conditions where at least one of them should be true.

7. DISTINCT Method

```
SELECT DISTINCT Grade FROM Students;
```

This query retrieves a list of unique grades from the `Students` table. The `DISTINCT` keyword is used to eliminate duplicate values.

8. AVG() Method

```
SELECT AVG(Marks) AS AverageMarks FROM Students;
```

This query calculates the average (`AVG()`) marks of all the students in the table. The `AVG()` function is used to calculate the mean of a numeric column.

9. COUNT() Method

SELECT COUNT(*) AS TotalStudents FROM Students;

This query returns the total number of students in the table using the `COUNT()` function. The `COUNT()` function counts the number of rows that match the query condition.

10. MAX() Method

SELECT MAX(Marks) AS HighestMarks FROM Students;

This query retrieves the highest marks achieved by any student. The `MAX()` function is used to find the maximum value in a numeric column.

11. SUM() Method

SELECT SUM(Marks) AS TotalMarks FROM Students;

This query calculates the sum (`SUM()`) of all the students' marks in the table. The `SUM()` function adds up the values in a numeric column.

Explanation of Each Query:

1. **Create Table**: Defines the structure of a table including column names and their data types.
2. **Insert**: Adds new rows (student records) to the table.
3. **Update**: Modifies existing records based on a condition.
4. **Alter Table**: Changes the structure of the table (e.g., adding or removing columns).
5. **WHERE with AND**: Filters records based on multiple conditions that must all be true.
6. **WHERE with OR**: Filters records where at least one of the conditions is true.
7. **DISTINCT**: Retrieves unique values from a column, removing duplicates.
8. **AVG()**: Calculates the average of values in a numeric column.
9. **COUNT()**: Counts the number of rows in a table that match the specified criteria.
10. **MAX()**: Retrieves the maximum value in a numeric column.
11. **SUM()**: Adds up the values in a numeric column.

These queries demonstrate how to manage and analyze student data in a database using SQL.