Date
Page

PRACTICE SET-1

Question 1 : Schema Definition
Create a Table query.

⇒
```
Create Table Students (
    StudentID INT Primary key, AUTO INCREMENT,
    FirstName VARCHAR(50) NOT NULL,
    LastName VARCHAR(50) NOT NULL,
    Age INT CHECK (Age >= 16),
    EnrollmentDate DATE DEFAULT CORRENT_DATE,
    Major VARCHAR (100)
);
```

Question 2 : Insert Data
Insert the following rows into student table:

```
INSERT INTO students (FirstName, LastName, Age,
                      EnrollmentDate, Major)
VALUES ("ALICE", "Johnson", 18, "2023-09-01", "ComputerScience"),
    ("Bob", "Smith", 20, "2022-06-15", "Mathematics"),
    ("Charlie", "Brown", 19, "2021-08-20", "Physics"),
    ("Daisy", "Carter", 21, "2023-01-10", "Biology"),
    ("Ethan", "Taylor", 22, "2023-03-25", "Chemistry");
```

Question 3 : Update Data

1. Change the major of the student with StudentID = 1 to "Data Science".

=> UPDATE students
~~SET Age = major~~ = SET Major = "Data Science"
WHERE StudentID = 1 ;

2 Increase the age of all students enrolled before "2023-01-01" by 1 year

=> UPDATE students
SET Age = Age + 1
WHERE "EnrollementDate < "2023-01-01" ;

3 Update the LastName of the students with FirstName = 'Daisy' to "Cooper".

=> UPDATE students
SET LastName = "Cooper"
WHERE FirstName = "Daisy" ;

4 Set the Major to "Undeclared" for all students younger than 20

=> UPDATE students
SET Major = "Undeclared"
WHERE Age < 20 ;

5 Update EnrollmentDate of the student with StudentID = 5 to "2024-01-01"

=> UPDATE students
SET EnrollmentDate = "2024-01-01"
WHERE StudentID = 5 ;

6 Set the Major to "Physics" for all students whose current major is "Biology"

```
=> UPDATE students
   SET Major = "Physics"
   WHERE "Major = "Biology";
```

7 Update the Age to 23 for all students with FirstName = "Charlie"

```
=> UPDATE students
   SET Age = 23
   WHERE FirstName = "Charlie";
```

8 Change the LastName of the student with Major = 'Mathematics' to "Williams".

```
=> UPDATE students
   SET LastName = "Williams"
   WHERE Major = "Mathematics";
```

9 Update the FirstName of the younget student to "Alex".

```
=> UPDATE Students
   SET FirstName = "Alex"
   WHERE Age = (SELECT MIN(Age) FROM students);
```

10 Set the Age to NULL for all students with Major = 'Undeclared'.

```
=> UPDATE students
   SET Age = "NULL"
   WHERE Major = "Undeclared";
```

11 Set the Major to DBMS for all youngest students.

```
UPDATE students
   SET Major = "DBMS"
   WHERE ~~Major~~ Age = (SELECT MIN(Age) FROM students);
```

Date
Page

Question 4 : Delete Records

1. Perform the following deletions on the students table

1. Delete the record of the student with StudentID = 3.

=> DELETE FROM students
WHERE StudentID = '1';

2. Remove all students with the Major as "Undeclared".

=> DELETE FROM students
WHERE Major = "Undeclared";

3. Delete all students who enrolled after "2023-01-01".

=> DELETE FROM Students
WHERE EnrollmentDate > "2023-01-01";

4. Remove students who are older than 21.

=> DELETE FROM students
WHERE Age > 21;

5 Delete the record of the student with FirstName = 'Ethan' and LastName = 'Taylor'

=> DELETE FROM students
WHERE FirstName = "Ethan" AND LastName = "Taylor";

6 Delete all students with Age = NULL

=> DELETE FROM students
WHERE Age IS NULL;

7 Remove all students with LastName starting with the letter "C"

=> DELETE FROM students
WHERE LastName LIKE "C%";

8 Delete all students where EnrollmentDate is earlier than "2022-01-01

=> DELETE FROM students
WHERE ~~Age~~ EnrollmentDate < "2022-01-01";

9 Remove all students who have "Physics" as their Major.

```
=> DELETE FROM students
   WHERE Major = "Physics";
```

10 Delete all records from the table without dropping the table itself.

```
=> DELETE FROM Students;
```

Practice set-2

1. Create Table with name Customers

```
=> CREATE TABLE Customers (
      CustomerID INT Primary Key, Auto Increament,
      FirstName VARCHAR(50) =;
      LastName VARCHAR(50),
      Email VARCHAR(100), Unique,
      PhoneNumber VARCHAR(15),
      Address VARCHAR(200));
```

2. Create Table with name Accounts

```
=> CREATE TABLE Accounts (
      AccountNumber INT Primary Key,
      CustomerID INT,
      AccountType VARCHAR(20) CHECK(AccountType
                  IN("Saving", "Checking")),
      Balance DECIMAL(15, 2),
      DateCreated DATE
      );
```

3 Insert the data into Customers table

=> INSERT INTO Customers (FirstName, LastName, E-mail, PhoneNumber, Address)
VALUES ("John", "Doe", "john.doe@gmail.com", "1234567890", "123 Main St, Cityville"),
("Jane", "Smith", "jane.smith@email.com", "0987654321", "456 Elm St, Townville"),
("Mike", "Johnson", "mike.johnson@email.com", "1122334", "7890 Oak St, Villageville");

4 Insert the data into Accounts table

=> INSERT INTO Accounts (AccountNumber, CustomerID, AccountType, Balance, DateCreated)
VALUES (1001, 1, "Saving", 5000.00, "2023-01-15"),
(1002, 1, "Checking", 1500.00, "2023-02-20"),
(1003, 2, "Saving", 2000.00, "2023-03-01"),
(1004, 3, "Checking", 3000.00, "2023-03-10");

5 Update Data

(i) Update the balance of the account with AccountNumber 1001 to 5500.00.

=> UPDATE Accounts
SET Balance = 5500.00
WHERE AccountNumber = 1001;

(ii) Update the E-mail of the customer with customerID 2 to jane.smith@newdomain.com.

=> UPDATE Customers
SET Email = "jane.smith@newdomain.com"
WHERE CustomerID = 2;

(iii) Increase the balance by 10% for all customers with a Saving account type

=> UPDATE Accounts
SET Balance = Balance + (10 * Balance) / 100
WHERE AccountType = "Saving";

6 SELECT Queries

(i) Write a query to list the CustomerID, FirstName, lastName, and Balance of all customers who have a saving acccount typ.

=> SELECT C.CustomerID, C.FirstName, C.lastName, A.Balance
FROM Customers C
JOIN Accounts A ON C.CustomerID = A.CustomerID
WHERE A.AccountType = "Saving";

(ii) Find all customers who have a balance greater than 3000.00 and their Account Type is Checking.

=> SELECT C.CustomerID, C.FirstName, C.lastName, A.Balance
FROM Customers C
JOIN Accounts A ON C.CustomerID = A.CustomerID
WHERE A.Balance > 3000.00 AND A.AccountType = "Checking";

(iii) List all accounts with a balance less than 2000.00, along with the CustomerID, AccountNumber and AccountTyp e.

=> SELECT CustomerID, AccountNumber, AccountType
FROM Accounts
WHERE Balance < 2000.00;

7 Deleting Data

(i) Delete the account with AccountNumber 1002.

=> DELETE FROM Accounts
WHERE AccountNumber = 1002;

(ii) Delete all customers whose PhoneNumber starts with "123"

=> DELETE FROM Customers
WHERE PhoneNumber LIKE '1234.';

(iii) Delete Accounts Created Before '2023-02-01'

=> DELETE FROM Accounts
WHERE DateCreated < '2023-02-01';

8 JOIN Queries

(i) Find FirstName, LastName, and AccountType of Customers with BalanceGreaterThan2000

=> SELECT C.FirstName, C.LastName, A.AccountType
FROM Customers C
JOIN Accounts A ON C.CustomerId = A.CustomerId
WHERE A.Balance > 2000;

(ii) Get Total Balance of Saving Accounts.

=> SELECT AccountType, SUM(Balance) As TotalBalance
FROM Accounts
WHERE AccountType = "Saving"
GROUP BY AccountType;

(iii) Write a query to show customer's FirstName, LastName, AccountNumber, and Balance from the Customers and Accounts tables. Ensure that it shows all customers, including those without an account.

=> SELECT C.FirstName, C.LastName, A.AccountNumber, A.Balance
FROM Customers C
LEFT JOIN Accounts A ON C.CustomerID = A.CustomerID;

9 Constraints and Validation

(i) Modify the Accounts table to ensure that the Balance Column must never be negative

=> ALTER TABLE Accounts
ADD CONSTRAINT ~~ck~~ Balance CHECK (Balance >= 0);

(ii) Modify the ~~Account~~ Customers table to ensure that the Email column must be unique and not null.

=> ALTER TABLE Customers
MODIFY COLUMN Email VARCHAR(100) NOT NULL UNIQUE;

(iii) Add a foreign key constraint to the Accounts table for the CustomerID column, linking it to the CustomerID column of the Customers table

=> ALTER TABLE Accounts
ADD CONSTRAINT fk_Customer FOREIGN KEY (CustomerID)
REFERENCES Customers (CustomerID);

v) Create Check Constraint for AccountType

=> ALTER TABLE Accounts
ADD CONSTRAINT check_account_type CHECK
(AccountType IN ('Saving', 'Checking'));

10. Complex Queries

(i) Write a query to show the customer with the highest balance across all account types.

```
=> SELECT C.FirstName, C.LastName, A.AccountNumber,
                                   Max(A.Balance) As highBalance
   FROM Customers C
   JOIN AccountsA ON C.CustomerID = A.CustomerID)
   GROUP BY C.CustomerID)
   ORDER BY highBalance DESC
   LIMIT 1;
```

(ii) Create a report that shows the total balance for each customer, including their FirstName and LastName. If a customer has multiple accounts, the total should be the sum of all their account balances

```
=> SELECT C.FirstName, C.LastName, SUM(A.Balance) As Total
   FROM Customers C
   JOIN Accounts A ON C.CustomerID = A.CustomerID)
   GROUP BY C.CustomerID), C.FirstName, C.LastName;
```

11 Aggregation

(i) Find the average balance of all accounts in the Banking system

```
=> SELECT AVG(Balance) As AverageBalance
   FROM Accounts;
```

(ii) Write a query to count the total number of saving accounts in sys...

```
=> SELECT COUNT(*) As TotalSavingAccounts
   FROM Accounts
   WHERE AccountType = "Saving";
```

PRACTICE SET 3 :

Date
Page

1. Create Table Query

```
=> CREATE TABLE Library (
        BOOKID INT Primary key, Auto-Increment,
        Title VARCHAR(500),
        Author VARCHAR(69),
        Publisher VARCHAR(15),
        Genre VARCHAR(20),
        PublishedYear ~~Integer~~ Integer,
        ISBN VARCHAR(29) UNIQUE,
        Pages INTEGER,
        CopiesAvailable INTEGER,
        Price DECIMAL. );
```

2 Inserting Data

```
=> ~~INSERT INTO Library(BOOKID, ...~~
=> INSERT INTO Library (Title, Author, Publisher, Genre,
        PublishedYear, ISBN, Pages, CopiesAvailable, Price)
VALUES("To Kill a Mockingbird", "Harper Lee", "J.B. Lippincott",
        "Fiction", 1960, "978-0061120084", 324, 5, 15.99 ),
("1984", "George Orwell", "Harvill Secker", "Dystopian",
  1949, "978-0451524935", 328, 2, 9.99 ),
("The Great Gatsby", "F. Scott Fitzgerald", "Scribner", "Fiction",
  1925, "978-0743273565", 180, 3, 10.99 ),
("The Catcher in the Rye", "J.D. Salinger", "Little, Brown", "Fiction",
  1951, "978-0316769488", 277, 4, 12.99 ),
("The Hobbit", "J.R.R. Tolkein", "HarperCollins", "Fantasy",
  1937, "978-0618968633", 310, 6, 13.99 );
```

3. Update Queries with Multiple Conditions

a) Update the Price of the book with ISBN '978-0451524935' to 11.99 if the Genre is 'Dystopian' and the Published Year is before 1950.

=> UPDATE Library
SET Price = 11.99
WHERE ISBN = "978-0451524935" AND Genre = "Dystopian" AND Published Year = "1950;

b) Update the number of Copies Available to 10 for all books with 'Fiction' genre and a published year after 1950.

=> UPDATE Library
SET CopiesAvailable = 10
WHERE Genre = "Fiction" AND PublishedYear = 1950;

c) Set the Price of books to 5% less for all books in the 'Fiction' genre with more than 300 pages

=> UPDATE Library
SET Price = Price - (5 * Price) / 100
WHERE Genre = "Fiction" AND ~~CopiesAvailable = "300";~~ Pages > 300;

D) Update the Pages of books to 350 if the CopiesAvailable are greater than 4 and the price is below 14.

=> UPDATE Library
SET Pages = 350
WHERE CopiesAvailable > 4 AND Price < 14;

e Increase the Price by 10% for all books in the 'Fantasy' genre published before 1950 and with less than 300 pages.

=> UPDATE Library
SET Price = Price + (10 * Price) / 100
WHERE Genre = "Fantasy" AND PublishedYear = 1950 AND Pages < 300;

f Set the CopiesAvailable to 0 for books where the Price is greater than 12 and the Genre is either 'Fiction' or 'Dystopian'

=> UPDATE Library
SET CopiesAvailable = 0
WHERE Price > 12 AND Genre IN ("Fiction", "Dystopian");

g Update the Published Year to 2020 for all books with 'Tolkein' in the author's name, where the price is greater than 10 and less than 15.

=> UPDATE Library
SET PublishedYear = 2020
WHERE Author LIKE "%. Tolkient.'' AND Price BETWEEN 10 AND 15;

h Set the Price to 8.99 for books with 'George Orwell' in the author's name and having more than 300 pages.

=> UPDATE Library
SET Price = 8.99
WHERE Author = "George Orwell" AND Pages > 300;

i Decrease the Price of all books in the 'Fiction' genre published before 1950 by 15%, if the copies Available are less than 15.

=> UPDATE Library
SET Price = Price - (Price * 15) / 100
WHERE Genre = "Fiction" AND Copies Available < 15;

j Update the Price of 'To Kill a Mocking bird' to 17.99, if the Published Year is 1960 and Copies Available is greater than 4.

=> UPDATE Library
SET Price = 17.99
WHERE Title = 'To Kill a Mocking bird' AND Published Year = 1960
AND Copies Available > 4;

4 Delete Queries with Multiple Conditions

a) Delete books with the ISBN '978-0451524935' and '978-0618-968633' if they belong to the 'Dystopian' genre and have more than 2 copies available.

=> DELETE FROM Library
WHERE ISBN IN ("978-0451524935", "978-0618968633") AND
Genre = "Dystopian" AND Copies Available > 2;

b) Delete all books published before 1950 that have a price lower than

=> DELETE FROM Library
WHERE Published Year < 1950 AND Price < 10;

c Delete books in the 'Fiction' genre with less than 3 copies, published before 1960.

=> DELETE FROM Library
WHERE Genre = "Fiction" AND Copies Available < 3 AND
Published Year < 1960;

d Delete all books with fewer than 200 pages and where the Published Year is greater than 1920 but less than 1960.

=> DELETE FROM Library
WHERE Pages < 200 and PublishedYear BETWEEN 1920 AND 1960;

e Delete books where the author's name is 'Harper Lee' and the price is less than 12.

=> DELETE FROM Library
WHERE Author = "Harperlee" and Price < 12.00;

f Delete all books where CopiesAvailable is 0 and the price is greater than 15.

=> DELETE FROM Library
WHERE CopiesAvailable = 0 AND Price > 15.00;

g Delete all books with ISBN containing '978-074', where the Genre is 'Fiction' and PublishedYear is before 1950.

=> DELETE FROM Library
WHERE ISBN LIKE "%.978-074" AND Genre = "Fiction" AND PublishedYear < 1950;

h Delete all books published after 2000 that have a price below 10 or greater than 15.

=> DELETE FROM Library
WHERE PublishedYear = 2000 AND
~~(Price BETWEEN 10 AND 15 OR Price > 15);~~
(Price < 10 OR Price > 15);

i Delete books in the 'Fantasy' genre where the number of copies Available is greater than 3 but less than 10.

=> DELETE FROM Library
WHERE Genre = 'Fantasy" AND CopiesAvailable BETWEEN 3 AND 10;

j Delete books with 'J.D. Salinger' as the author, and where the number of pages is less than 300 and the price is greater than 12.

=> DELETE FROM Library
WHERE Author = "J.D. Salinger" AND Pages < 300 AND Price > 12;

5 Select Queries with Multiple Conditions

a) Find books in the 'Fiction' genre that were published after 1950, have more than 200 pages and the price is between 10 and 15.

=> SELECT *
FROM Library
WHERE Genre = "Fiction" AND PublishedYear > 1950 AND Pages > 200
AND Price BETWEEN 10 AND 15;

6) List all books with more than 300 pages and a price greater than 12, excluding books published in the 'Fantasy' genre.

=> SELECT *
FROM Library
WHERE Pages > 300 AND Price > 12.00 , AND Genre != "Fantasy";

Get a list of books published between 1925 and 1950, price below 13, and have more than 3 copies.

=> SELECT *
FROM Library
WHERE PublishedYear BETWEEN 1925 AND 1950 AN Price < 13.00
AND CopiesAvailable > 3;

d Find books by 'Harperlee' or 'George Orwell', wher the PublishedYear is greater than 1950 and the price is morethan 10

=> SELECT *
FROM Library
WHERE (Author = "Harperlee" OR Author = "George Orwell")
AND PublishedYear > 1950 AND Price > 10;

e List books by 'J.R.R Tolkein' where Copies Available is greater than 5, and price is between 12 and 15.

=> SELECT *
FROM Library
WHERE Author = "J.R.R Tolkein" AND CopiesAvailable > 5
AND Price BETWEEN 12.00 AND 15.00;

f Show all books with 'Tolkien' in the author's name and PublishedYear befor 1940, where Copies between 4 and 6.

=> SELECT *
FROM Library
WHERE Author LIKE "%. Tolkein %." AND
PublishedYear < 1940 AND CopiesAvailable BETWEEN
4 AND 6;

g Find the book with the highest price and more than 300 pages published in the 'Dystopian' or 'Fantasy' Genre

=> SELECT *
FROM Library
WHERE Pages > 300 AND (Genre = "Dystopian" or Genre = "Fantasy")
ORDER BY Price DESC,
LIMIT 1;

h List books in 'Fiction' genre with more than 200 pages and Price between 10 and 20, Copies Available between 2 and 5:

=> SELECT *
FROM Library
WHERE Genre = "Fiction" AND Pages > 200 AND Price
BETWEEN 10 AND 20 AND CopiesAvailable BETWEEN
2 AND 5;

i Find all books published before 1950, Price < 15 and Copies < 3

=> SELECT *
FROM library
WHERE PublishedYear < 1950 AND Price < 15.00 AND
CopiesAvailable < 3;

j Show titles and authors of books with Price more than 12, Copies more than 4 and published after 1930

=> SELECT Title, Author
FROM library
WHERE Price > 12.00 AND CopiesAvailable > 4 AND
PublishedYear > 1930;