Name : Raval Shreya D.

Enrollment No : 2202031800034

Branch : CSE

DIVISION : Internship (B3)

‘C’

Practical-1

Aim: Write a C program to print the address of a variable using a pointer.

Code:

```
#include <stdio.h>

int main() {

int num = 42;

int *ptr = &num;

printf("The address of num is: %p\n", &num);

printf("The value of ptr is: %p ", ptr);

return 0;

}
```

Output:

The address of num is: 0x7fff5fbff7e8

The value of ptr is: 0x7fff5fbff7e8

Practical-2

Aim: Write a C program to create a Calculator using a pointer.

Code:

```
#include <stdio.h>

int main() {
   int num1, num2, result;
   char operator;
   int *p1, *p2;

   printf("Enter two numbers: ");
   scanf("%d %d", &num1, &num2);

   printf("Enter the operator (+, -, *, /): ");
   scanf(" %c", &operator);

   p1 = &num1;
   p2 = &num2;

   switch(operator) {
```

```
        case '+':
            result = *p1 + *p2;
            printf("%d + %d = %d\n", *p1, *p2, result);
            break;

        case '-':
            result = *p1 - *p2;
            printf("%d - %d = %d\n", *p1, *p2, result);
            break;

        case '*':
            result = *p1 * *p2;
            printf("%d * %d = %d\n", *p1, *p2, result);
            break;

        case '/':
            result = *p1 / *p2;
            printf("%d / %d = %d\n", *p1, *p2, result);
            break;

        default:
            printf("Invalid operator\n");
    }

    return 0;
}
```

Output :

```
Enter two numbers: 10  5
Enter the operator (+, -, *, /): +
```

10 + 5 = 15

Practical-3

Aim : Write a C program to swap the two values using call by value and call by reference

Code:

```
#include <stdio.h>

// Function to swap two values using call by value
void swap_by_value(int x, int y) {
   int temp = x;
   x = y;
   y = temp;
   printf("After swapping (call by value):\n x = %d, y = %d\n", x, y);
}

// Function to swap two values using call by reference
void swap_by_reference(int *x, int *y) {
   int temp = *x;
   *x = *y;
   *y = temp;
   printf("After swapping (call by reference):\n x = %d, y = %d\n", *x, *y);
}

int main() {
   int num1 = 10, num2 = 5;

   printf("Before swapping:\n x = %d, y = %d\n", num1, num2);

   // Swap by value
   swap_by_value(num1, num2);

   // Swap by reference
   swap_by_reference(&num1, &num2);

   return 0;
}
```

Output:

```
Before swapping: x = 5, y = 10
```

Values inside the swapByValue function: a = 10, b = 5

After swapping: x = 5, y = 10

Before swapping: x = 5, y = 10

Values inside the swapByReference function: a = 10, b = 5

After swapping: x = 10, y = 5

Practical-4

Aim : Define a structure type struct personal that would contain person name, Date of birth and age using this structure to read this information of 4 people and display the same.

Code :

```
 #include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct personal {
    char name[50];
    int birth_year;
    int birth_month;
    int birth_day;
    int age;
};

int main() {
    int i;
    struct personal people[4];

    for (i = 0; i < 4; i++) {
        printf("Enter the name of person %d: ", i+1);
```

```
        fgets(people[i].name, 50, stdin);
        strtok(people[i].name, "\n"); // remove trailing newline
        printf("Enter the birth year of person %d: ", i+1);
        scanf("%d", &people[i].birth_year);
        printf("Enter the birth month of person %d: ", i+1);
        scanf("%d", &people[i].birth_month);
        printf("Enter the birth day of person %d: ", i+1);
        scanf("%d", &people[i].birth_day);
        getchar(); // remove trailing newline
        people[i].age = 2023 - people[i].birth_year;
    }

    printf("\n");
    for (i = 0; i < 4; i++) {
        printf("Person %d:\n", i+1);
        printf("  Name: %s\n", people[i].name);
        printf("  Date of birth: %d-%d-%d\n", people[i].birth_year,
people[i].birth_month, people[i].birth_day);
        printf("  Age: %d\n", people[i].age);
    }

    return 0;
}
```

Output :

```
Enter the name of person 1: John Smith
Enter the birth year of person 1: 1990
Enter the birth month of person 1: 3
Enter the birth day of person 1: 14
Enter the name of person 2: Sarah Johnson
Enter the birth year of person 2: 1985
Enter the birth month of person 2: 9
Enter the birth day of person 2: 20
```

```
Enter the name of person 3: Alex Brown
Enter the birth year of person 3: 1995
Enter the birth month of person 3: 5
Enter the birth day of person 3: 3
Enter the name of person 4: Emily Davis
Enter the birth year of person 4: 1980
Enter the birth month of person 4: 12
Enter the birth day of person 4: 31

Person 1:
  Name: John Smith
  Date of birth: 1990-3-14
  Age: 33
Person 2:
  Name: Sarah Johnson
  Date of birth: 1985-9-20
  Age: 38
Person 3:
  Name: Alex Brown
  Date of birth: 1995-5-3
  Age: 28
Person 4:
  Name: Emily Davis
  Date of birth: 1980-12-31
  Age: 43
```

Practical-5

Aim : Write a C program to calculate the sum of n numbers entered by the user using dynamic memory allocation.

Code:

```
#include <stdio.h>

#include <stdlib.h>


int main() {

   int n, i, *ptr, sum = 0;


   printf("Enter the number of integers you want to sum: ");
```

```
    scanf("%d", &n);

    ptr = (int*) malloc(n * sizeof(int)); // allocate memory for n integers

    if (ptr == NULL) { // check if memory allocation was successful
        printf("Memory allocation failed. Exiting program.\n");
        return 1;
    }

    printf("Enter %d integers:\n", n);
    for (i = 0; i < n; i++) {
        scanf("%d", &ptr[i]);
        sum += ptr[i];
    }

    printf("The sum of the %d integers is %d\n", n, sum);

    free(ptr); // free allocated memory

    return 0;
}
```

Output :

```
Enter the number of integers you want to sum: 5
Enter 5 integers:
1
2
3
4
5
The sum of the 5 integers is 15
```

Practical-6

Aim : A file named “New” contains a series of integer numbers. Write a c program to read all numbers from a file and then copy all odd numbers into a file named “odd” and write all even numbers into a file named “even”. Then display the values of files odd and even on the screen.

Code :

```
#include <stdio.h>


  int main() {
    FILE *fp, *fp_odd, *fp_even; // file pointers for input and output files
    int num;

    fp = fopen("New.txt", "r"); // open input file for reading

    if (fp == NULL) { // check if file was opened successfully
      printf("Error opening file New.txt\n");
      return 1;
    }

    fp_odd = fopen("odd.txt", "w"); // open output file for odd numbers
    fp_even = fopen("even.txt", "w"); // open output file for even numbers

    if (fp_odd == NULL || fp_even == NULL) { // check if files were opened
  successfully
      printf("Error opening output files\n");
      return 1;
    }

    while (fscanf(fp, "%d", &num) != EOF) { // read integers from input file
```

```
        if (num % 2 == 0) {
            fprintf(fp_even, "%d ", num); // write even numbers to output file
        } else {
            fprintf(fp_odd, "%d ", num); // write odd numbers to output file
        }
    }

    fclose(fp); // close input file
    fclose(fp_odd); // close output file for odd numbers
    fclose(fp_even); // close output file for even numbers

    printf("Contents of odd.txt:\n");
    fp_odd = fopen("odd.txt", "r"); // open output file for reading

    while (fscanf(fp_odd, "%d", &num) != EOF) { // read integers from output file for odd numbers
        printf("%d ", num);
    }

    printf("\nContents of even.txt:\n");
    fp_even = fopen("even.txt", "r"); // open output file for reading

    while (fscanf(fp_even, "%d", &num) != EOF) { // read integers from output file for even numbers
        printf("%d ", num);
    }

    fclose(fp_odd); // close output file for odd numbers
    fclose(fp_even); // close output file for even numbers
```

```
    return 0;
}
```

Output :

```
Contents of odd.txt:
1 3 5 7 9
Contents of even.txt:
2 4 6 8 10
```

‘C++’

Practical-7

Aim : Write a C++ program to Check if the number is prime or not using a function.

Code :

```
#include <iostream>
using namespace std;

// Function to check if a number is prime or not
bool isPrime(int num) {
    if (num <= 1) {
        return false;
    }

    for (int i = 2; i <= num/2; i++) {
        if (num % i == 0) {
            return false;
        }
    }

    return true;
}

int main() {
    int num;

    cout << "Enter a number: ";
    cin >> num;

    if (isPrime(num)) {
        cout << num << " is a prime number.";
```

```
    } else {
        cout << num << " is not a prime number.";
    }

    return 0;
}
```

Output :

```
Enter a number: 17
17 is a prime number.
```

Practical-8

Aim : Write a C++ program that prompts the user to enter a letter and check whether a letter is a vowel or constant.

Code :

```
#include <iostream>
using namespace std;

int main() {
    char letter;

    cout << "Enter a letter: ";
    cin >> letter;

    switch (letter) {
        case 'a':
        case 'e':
        case 'i':
        case 'o':
        case 'u':
        case 'A':
        case 'E':
        case 'I':
        case 'O':
        case 'U':
            cout << letter << " is a vowel.";
            break;
        default:
            cout << letter << " is a consonant.";
            break;
```

```
    }

    return 0;
}
```

Output :

```
Enter a letter: o
o is a vowel.
```

Practical-9

Aim : Write a C++ program to demonstrate the concept of constructor and destructor.

Code :

```
#include <iostream>
using namespace std;

class MyClass {
public:
    // Constructor
    MyClass() {
        cout << "Constructor called." << endl;
    }

    // Destructor
    ~MyClass() {
        cout << "Destructor called." << endl;
    }
};

int main() {
    cout << "Creating an object of MyClass." << endl;
    MyClass obj;

    cout << "Exiting main function." << endl;
    return 0;
}
```

Output :

```
Creating an object of MyClass.
Constructor called.
Exiting main function.
Destructor called.
```

Explanation :

We define a class **`MyClass`** that has a constructor and a destructor.
The constructor is called when an object of the class is created. In this case, the constructor simply prints the message "Constructor called." to the console.
The destructor is called when an object of the class is destroyed. In this case, the destructor simply prints the message "Destructor called." to the console.
In the **`main`** function, we create an object of the **`MyClass`** class using the default constructor. After creating the object, we print the message "Exiting main function." to the console and return 0.
When the **`main`** function exits, the object of **`MyClass`** is destroyed, and the destructor is called.

Practical-10

Aim : Create a class student that stores roll_no, name. Create a class test that stores marks obtained in five subjects. Class result derived from student and test contains the total marks and percentage obtained in test. Input and display information of a student.

Code :

```
#include <iostream>
#include <string>
using namespace std;

class Student {
protected:
   int roll_no;
   string name;
public:
   void get_data() {
      cout << "Enter roll number: ";
      cin >> roll_no;
      cout << "Enter name: ";
      cin.ignore();
      getline(cin, name);
   }
   void display_data() {
      cout << "Roll number: " << roll_no << endl;
      cout << "Name: " << name << endl;
   }
};

class Test {
protected:
   float marks[5];
public:
```

```
    void get_marks() {
        cout << "Enter marks obtained in five subjects:" << endl;
        for (int i = 0; i < 5; i++) {
            cout << "Subject " << i+1 << ": ";
            cin >> marks[i];
        }
    }
    void display_marks() {
        cout << "Marks obtained in five subjects:" << endl;
        for (int i = 0; i < 5; i++) {
            cout << "Subject " << i+1 << ": " << marks[i] << endl;
        }
    }
};

class Result : public Student, public Test {
private:
    float total_marks;
    float percentage;
public:
    void calculate_result() {
        total_marks = 0;
        for (int i = 0; i < 5; i++) {
            total_marks += marks[i];
        }
        percentage = total_marks / 5;
    }
    void display_result() {
        cout << "Total marks: " << total_marks << endl;
        cout << "Percentage: " << percentage << "%" << endl;
    }
};

int main() {
    Result r;
    r.get_data();
    r.get_marks();
    r.calculate_result();
    cout << endl;
    r.display_data();
    r.display_marks();
    r.display_result();

    return 0;
```

```
    }
```

Output : Enter roll number: 123
Enter name: John Doe
Enter marks obtained in five subjects:
Subject 1: 85
Subject 2: 92
Subject 3: 78
Subject 4: 90
Subject 5: 88

Roll number: 123
Name: John Doe
Marks obtained in five subjects:
Subject 1: 85
Subject 2: 92
Subject 3: 78
Subject 4: 90
Subject 5: 88
Total marks: 433
Percentage: 86.6%

Practical-11

Aim : Write a C++ program to overload binary + operator.

Code :

```
#include <iostream>
using namespace std;

class Number {
private:
   int value;
public:
   Number() {
      value = 0;
   }
   Number(int val) {
      value = val;
   }
   int get_value() {
      return value;
   }
   Number operator+(Number& num) {
      int sum = value + num.get_value();
      return Number(sum);
```

```
    }
};

int main() {
    Number num1(5);
    Number num2(10);
    Number sum = num1 + num2;
    cout << "Sum of " << num1.get_value() << " and " << num2.get_value() << " is " <<
sum.get_value() << endl;
    return 0;
}
```

Output : Sum of 5 and 10 is 15

Practical-12

Aim : Create a base class called 'SHAPE' having two data members of type double, member function get_data( ) to initialize base class data members, pure virtual member function display_area( ) to compute and display the area of the geometrical object. Derive two specific classes 'TRIANGLE' and 'RECTANGLE' from the base class. Using these three classes design a program that will accept dimension of a triangle / rectangle interactively and display the area.

Code :

```
#include <iostream>
using namespace std;

class SHAPE {
protected:
    double height;
    double width;
public:
    void get_data() {
        cout << "Enter height: ";
        cin >> height;
        cout << "Enter width: ";
        cin >> width;
    }
    virtual void display_area() = 0;
};

class TRIANGLE : public SHAPE {
public:
    void display_area() {
```

```
        double area = 0.5 * height * width;
        cout << "Area of triangle: " << area << endl;
    }
};

class RECTANGLE : public SHAPE {
public:
    void display_area() {
        double area = height * width;
        cout << "Area of rectangle: " << area << endl;
    }
};

int main() {
    SHAPE* shape_ptr;
    TRIANGLE;
    RECTANGLE;
    char choice;
    do {
        cout << "Enter T for Triangle, R for Rectangle: ";
        cin >> choice;
        if (choice == 'T') {
            shape_ptr = &triangle;
            cout << "Enter dimensions of triangle:" << endl;
            shape_ptr->get_data();
            shape_ptr->display_area();
        }
        else if (choice == 'R') {
            shape_ptr = &rectangle;
            cout << "Enter dimensions of rectangle:" << endl;
            shape_ptr->get_data();
            shape_ptr->display_area();
        }
        else {
            cout << "Invalid choice" << endl;
        }
        cout << "Do you want to continue? (Y/N): ";
        cin >> choice;
    } while (choice == 'Y' || choice == 'y');
    return 0;
}
```

Output :

```
Enter T for Triangle, R for Rectangle: T
```

Enter dimensions of triangle:
Enter height: 10
Enter width: 5
Area of triangle: 25
Do you want to continue? (Y/N): y
Enter T for Triangle, R for Rectangle: R
Enter dimensions of rectangle:
Enter height: 8
Enter width: 6
Area of rectangle: 48
Do you want to continue? (Y/N): n

'DBMS'

Practical-13

Aim : To study DDL-create and DML-insert commands.

Create following Table
Job (job_id, job_title, min_sal, max_sal)

Code :

```
CREATE TABLE Job (
job_id INT PRIMARY KEY,
job_title VARCHAR(50),
min_sal FLOAT,
max_sal FLOAT
);

INSERT INTO Job (job_id, job_title, min_sal, max_sal)
VALUES (1, 'Manager', 50000, 100000);

INSERT INTO Job (job_id, job_title, min_sal, max_sal)
VALUES (2, 'Engineer', 40000, 80000);

INSERT INTO Job (job_id, job_title, min_sal, max_sal)
VALUES (3, 'Analyst', 30000, 60000);
```

Output :

job_id	job_title	min_sal	max_sal
1	Manager	50000	100000

job_id	job_title	min_sal	max_sal
2	Engineer	40000	80000
3	Analyst	30000	60000

Practical-14

(i) Aim :

COLUMN NAME	DATA TYPE
job_id	Varchar(15)
job_title	Varchar(30)
min_sal	Int
max_sal	Int

Code :

```
CREATE TABLE Job (
job_id VARCHAR(15) PRIMARY KEY,
job_title VARCHAR(30),
min_sal INT,
max_sal INT
);
```

Output : Same as aim

(ii) Aim : Employee (emp_no, emp_name, emp_sal, emp_comm, dept_no)

Code :

```
CREATE TABLE Employee (
emp_no INT,
emp_name VARCHAR(30),
emp_sal DECIMAL(8, 2),
```

```
  emp_comm DECIMAL(6, 1),
  dept_no INT
);
```

Output :

COLUMN NAME	DATA TYPE
emp_no	Int
emp_name	Varchar(30)
emp_sal	decimal(8,2)
emp_comm	decimal(6,1)
dept_no	Int

(iii) Aim : deposit(a_no,cname,bname,amount,a_date)

Code :
```
CREATE TABLE deposit (
    a_no INT IDENTITY NOT NULL,
    cname VARCHAR(50) NOT NULL,
    bname VARCHAR(30) NOT NULL,
    amount DECIMAL(4,2) NOT NULL,
    a_date DATE NOT NULL,
    PRIMARY KEY (a_no, a_date)
);
```

Output :

COLUMN NAME	DATA TYPE
a_no	Int,identity
Cname	Varchar(50)
Bname	Varchar(30)
Amount	Decimal(4,2)
a_date	Date

(iv)Aim : borrow(loanno,cname,bname,amount)

Code :
```
CREATE TABLE borrow (
    loanno INT NOT NULL,
    cname VARCHAR(25) NOT NULL,
    bname VARCHAR(20) NOT NULL,
    amount DECIMAL(6,2) NOT NULL,
    PRIMARY KEY (loanno)
);
```

Output :

COLUMN NAME	DATA TYPE
Loanno	Int
Cname	Varchar(25)
Bname	Varchar(20)
Amount	Decimal(6,2)

Practical-15

Aim : Write SQL queries to insert following data into tables

(i)

emp_n	emp_name	emp_sal	emp_comm	dept _no
101	Smith	800		20
102	Snehal	1600	300	25
103	Adama	1100	0	20
104	Aman	3000		15

105	Anita	5000	50000	10
106	Sneha	2450	24500	10
107	Anamika	2975		30

(ii) Insert following values in the table **Job.**

job_id	**job_name**	**min_sal**	**max_sal**
IT_PROG	Programmer	4000	10000
MK_MGR	Marketing manager	9000	15000
FI_MGR	Finance manager	8200	12000
FI_ACC	Account	4200	9000
LEC	Lecturer	6000	17000
COMP_OP	Computer Operator	1500	3000

(iii) Insert following values in the table **deposit**.

A_no	**cname**	**Bname**	**Amount**	**date**
101	Anil	Andheri	7000	01-jan-06
102	sunil	Virar	5000	15-jul-06
103	jay	Villeparle	6500	12-mar-06
104	vijay	Andheri	8000	17-sep-06
105	keyur	Dadar	7500	19-nov-06
106	mayur	Borivali	5500	21-dec-06

Code :

```
(i) INSERT INTO employee (emp_n, emp_name, emp_sal, emp_comm, dept_no)VALUES
    (101, 'Smith', 800, 20, 20),
    (102, 'Snehal', 1600, 300, 25),
    (103, 'Adama', 1100, 0, 20),
    (104, 'Aman', 3000, 15, NULL),
    (105, 'Anita', 5000, 50000, 10),
    (106, 'Sneha', 2450, 24500, 10),
    (107, 'Anamika', 2975, 30, NULL);
   (ii) INSERT INTO Job (job_id, job_name, min_sal, max_sal) VALUES
    ('IT_PROG', 'Programmer', 4000, 10000),
    ('MK_MGR', 'Marketing manager', 9000, 15000),
    ('FI_MGR', 'Finance manager', 8200, 12000),
```

('FI_ACC', 'Account', 4200, 9000),
('LEC', 'Lecturer', 6000, 17000),
('COMP_OP', 'Computer Operator', 1500, 3000);

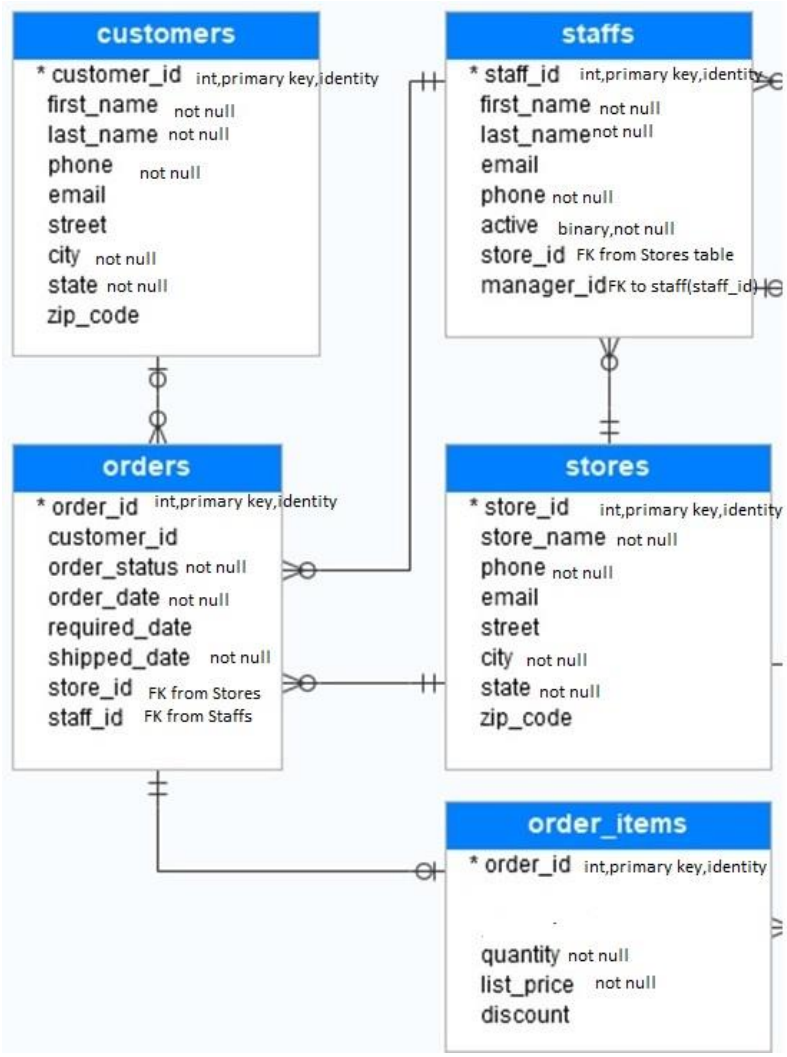
(iii) INSERT INTO deposit (a_no, cname, bname, amount, a_date) VALUES
(101, 'Anil', 'andheri', 7000, '2006-01-01'),
(102, 'sunil', 'virar', 5000, '2006-07-15'),
(103, 'jay', 'villeparle', 6500, '2006-03-12'),
(104, 'vijay', 'andheri', 8000, '2006-09-17'),
(105, 'keyur', 'dadar', 7500, '2006-11-19'),
(106, 'mayur', 'borivali', 5500, '2006-12-21');

Output : Same as represented in aim

Practical-16

Aim : Write the SQL queries to provide constraints on given tables.

Create A Database Sales and Write SQL Queries to create following tables with all constrains mentioned in image

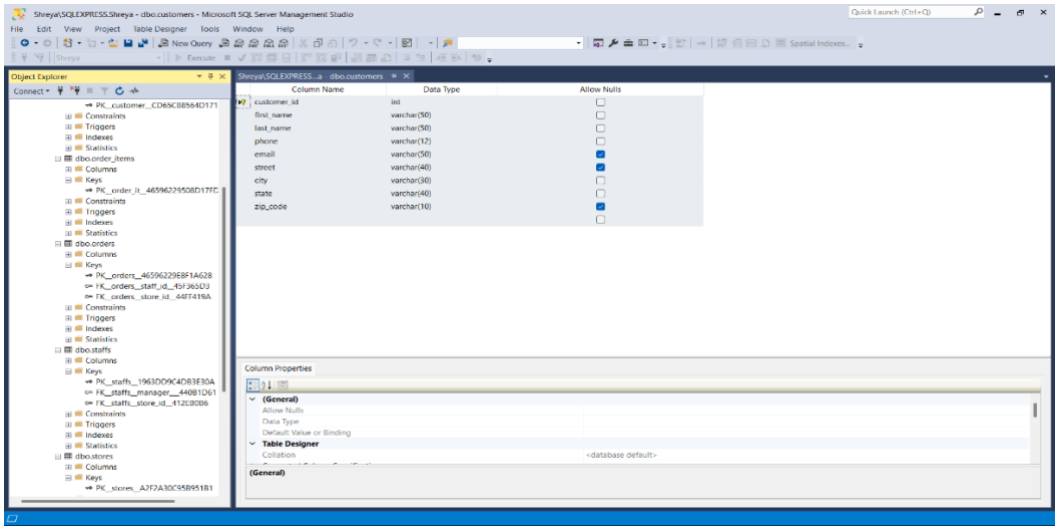


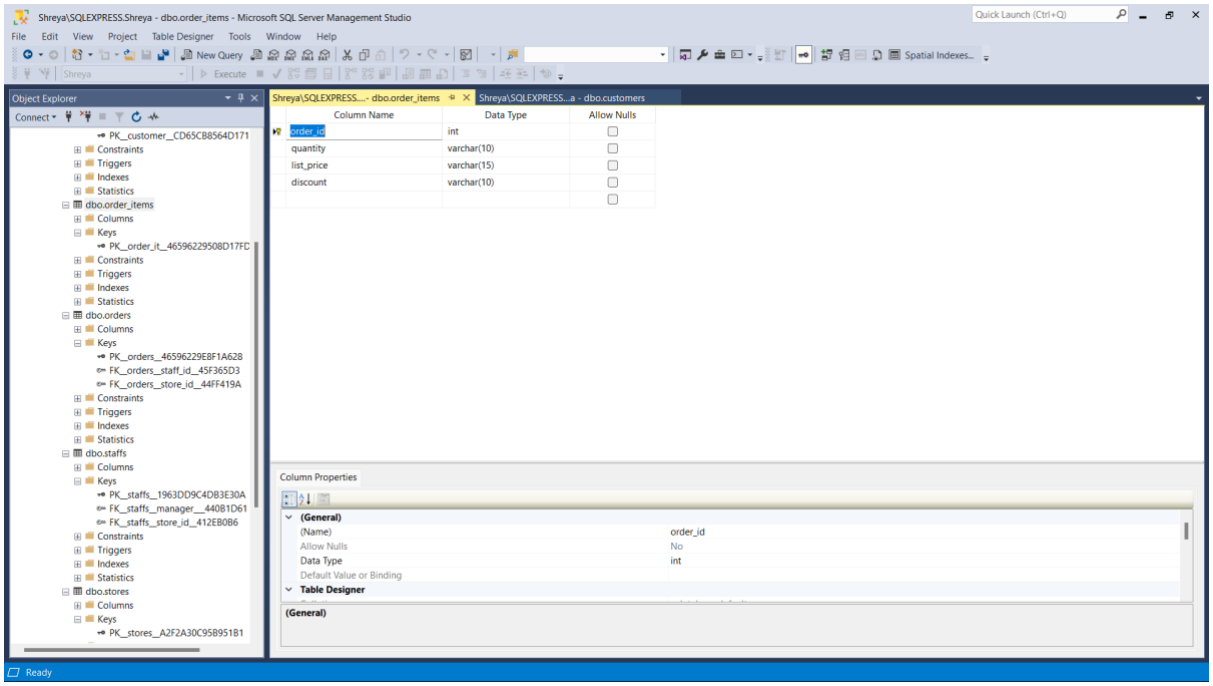
Code :

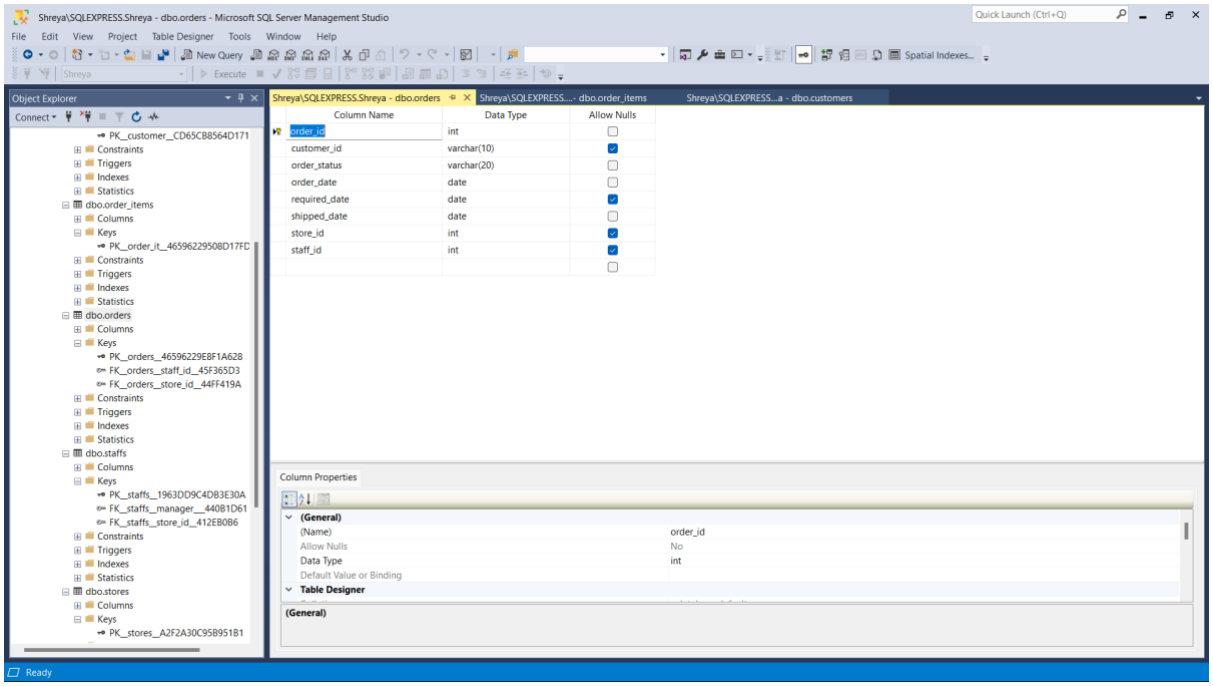
```
 CREATE TABLE customers(
customer_id int primary key identity,
first_name varchar(50) not null,
last_name varchar(50) not null,
phone varchar(12) not null,
email varchar(50) ,
street varchar(40),
city varchar(30) not null,
state varchar(40) not null,
zip_code varchar(10)
)

CREATE TABLE staffs(
staff_id int primary key identity,
first_name varchar(50) not null,
last_name varchar(50) not null,
email varchar(50) ,
phone varchar(12) not null,
active binary not null,
store_id int,
manager_id int
)

CREATE TABLE orders(
order_id int primary key identity,
customer_id varchar(10) ,
order_status varchar(20) not null,
order_date date not null,
required_date date ,
shipped_date date not null,
store_id int,
staff_id int
)

CREATE TABLE stores(
store_id int primary key identity,
store_name varchar(20) not null,
phone varchar(15) not null,
email varchar(30) ,
street varchar(15),
city varchar(15) not null,
state varchar(20) not null,
zip_code varchar(10)
)

CREATE TABLE order_items(
order_id int primary key identity,
quantity varchar(10) not null,
list_price varchar(15) not null,
discount varchar(10) not null
)

ALTER TABLE staffs
ADD FOREIGN KEY (store_id) REFERENCES stores(store_id);

ALTER TABLE staffs
ADD FOREIGN KEY (manager_id) REFERENCES staffs(staff_id);

ALTER TABLE orders
ADD FOREIGN KEY (store_id) REFERENCES stores(staff_id);
```

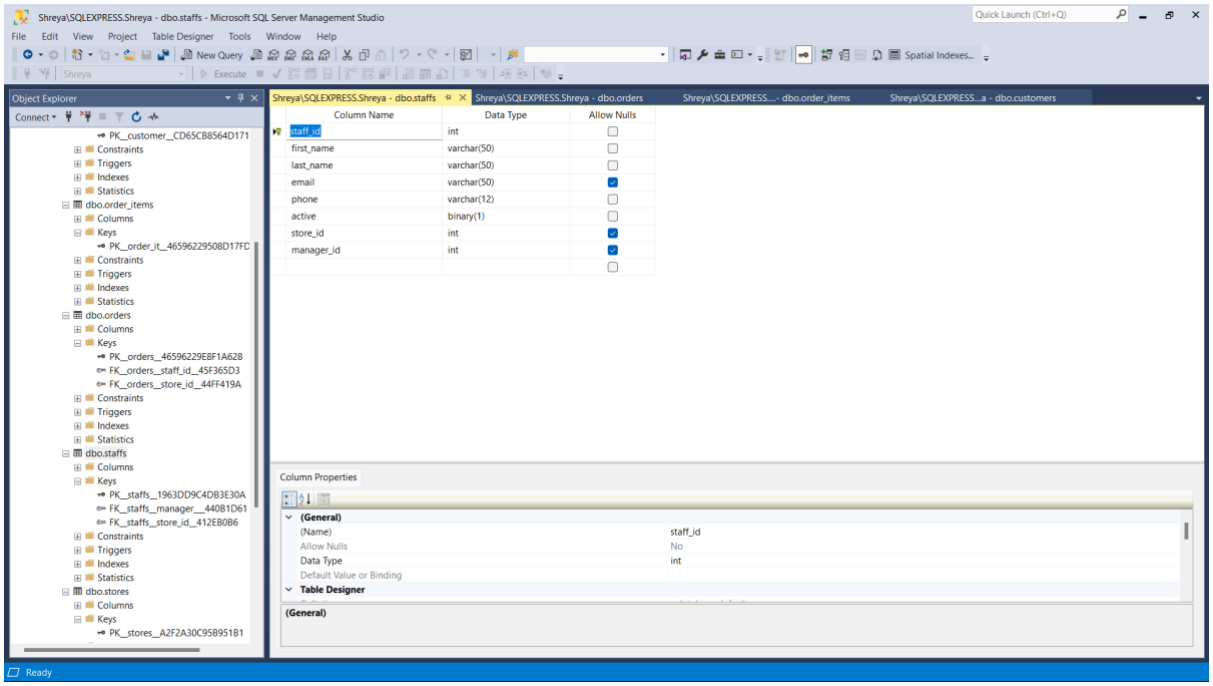
```
ALTER TABLE orders
ADD FOREIGN KEY (staff_id) REFERENCES staffs(staff_id);
```

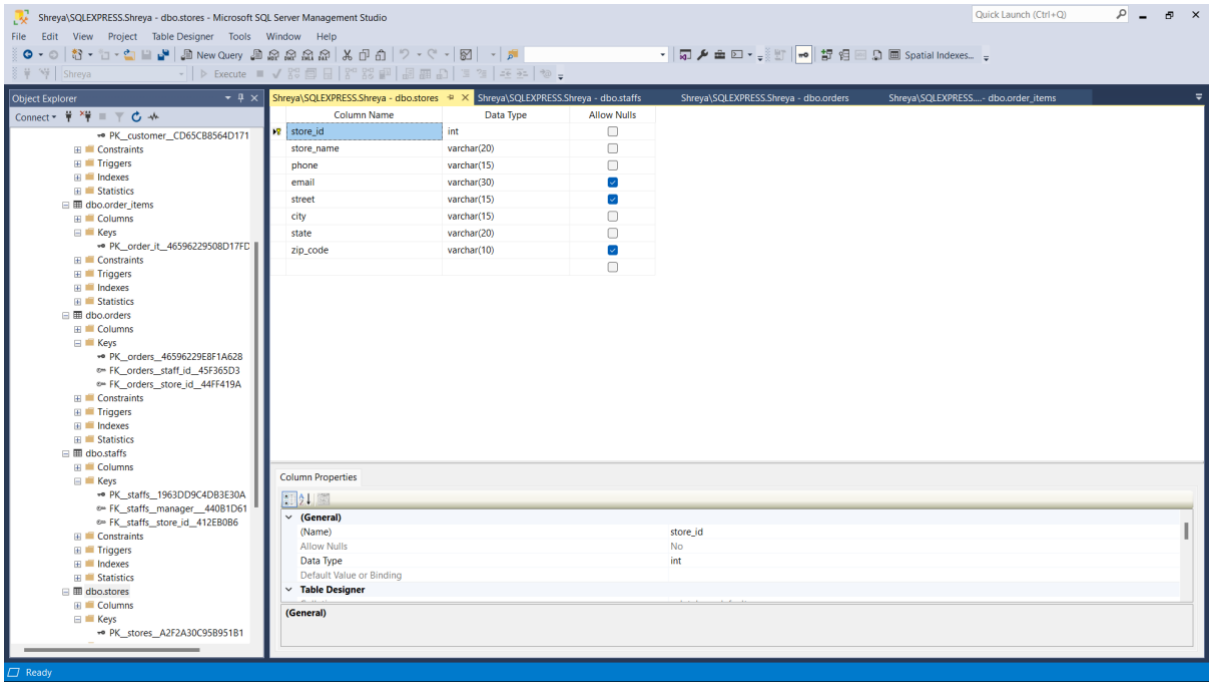
Output :






Shreya\SQLEXPRESS.Shreya - dbo.orders - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Project Table Designer Tools Window Help
New Query
Spatial Indexes...
Execute
Object Explorer
Connect
PK__customer__CD65CB8564D171
Constraints
Triggers
Indexes
Statistics
dbo.order_items
Columns
Keys
PK__order_it__4659622950BD17FC
dbo.orders
FK__orders__staff_id__45F365D3
FK__orders__store_id__44FF419A
dbo.staffs
PK__staffs__1963DD9C4DB3E30A
FK__staffs__manager__440B1D61
FK__staffs__store_id__412EB0B6
dbo.stores
PK__stores__A2F2A30C95B951B1
Shreya\SQLEXPRESS.Shreya - dbo.orders
Shreya\SQLEXPRESS....- dbo.order_items
Shreya\SQLEXPRESS....a - dbo.customers
Column Name
Data Type
Allow Nulls
order_id
int
customer_id
varchar(10)
order_status
varchar(20)
order_date
date
required_date
date
shipped_date
date
store_id
int
staff_id
int
Column Properties
(General)
(Name)
order_id
Allow Nulls
No
Data Type
int
Default Value or Binding
Table Designer
(General)
Ready


Shreya\SQLEXPRESS.Shreya - dbo.staffs - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Project Table Designer Tools Window Help
New Query
Spatial Indexes...
Execute
Object Explorer
Connect
PK__customer__CD65CB8564D171
Constraints
Triggers
Indexes
Statistics
dbo.order_items
Columns
Keys
PK__order_it__4659622950BD17FC
dbo.orders
FK__orders__staff_id__45F365D3
FK__orders__store_id__44FF419A
dbo.staffs
PK__staffs__1963DD9C4DB3E30A
FK__staffs__manager__440B1D61
FK__staffs__store_id__412EB0B6
dbo.stores
PK__stores__A2F2A30C95B951B1
Shreya\SQLEXPRESS.Shreya - dbo.staffs
Shreya\SQLEXPRESS.Shreya - dbo.orders
Shreya\SQLEXPRESS....- dbo.order_items
Shreya\SQLEXPRESS....a - dbo.customers
Column Name
Data Type
Allow Nulls
staff_id
int
first_name
varchar(50)
last_name
varchar(50)
email
varchar(50)
phone
varchar(12)
active
binary(1)
store_id
int
manager_id
int
Column Properties
(General)
(Name)
staff_id
Allow Nulls
No
Data Type
int
Default Value or Binding
Table Designer
(General)
Ready



Practical-17

Aim :

(i) Write the SQL queries to perform various aggregate functions on table data.

1. List total deposit from deposit.
2. List total amount from andheri branch
3. Count total number of customers
4. Count total number of customer's cities.
5. Update the value dept_no to 10 where second character of emp. name is 'm'.
6. Update the value of employee name whose employee number is 103.
7. Write a query to display the current date. Label the column Date
8. For each employee, display the employee number, salary, and salary increased by 15% and expressed as a whole number. Label the column New Salary
9. Modify your previous query to add a column that subtracts the old salary from the new salary. Label the column Increment.

(ii) Write the SQL queries to perform numeric, date and String functions.

1. Retrieve all data from employee, jobs and deposit.
2. Give details of account no. and deposited rupees of customers having account opened between dates 01-01-06 and 25-07-06.
3. Display all jobs with minimum salary is greater than 4000.

4. Display name and salary of employee whose department no is 20. Give alias name to name of employee.
5. Display employee no,name and department details of those employee whose department lies in(10,20)
6. Display all employee whose name start with ‘A’ and third character is ‘ ‘a’.
7. Display name, number and salary of those employees whose name is 5 characters long and first three characters are ‘Ani’.
8. Display the non-null values of employees and also employee name second character should be ‘n’ and string should be 5 character long.
9. Display the null values of employee and also employee name’s third character should be ‘a’.

Code :

(i) 1. List total deposit from deposit.

```
SELECT SUM(Amount) as Total_Deposit FROM deposit;
```

2. List total amount from andheri branch.

```
SELECT SUM(Amount) as Total_Amount FROM deposit WHERE
Bname='andheri';
```

3. Count total number of customers

```
SELECT COUNT(DISTINCT  bname) AS  total_cities FROM deposit;
```

4. Count total number of customer’s cities.

```
SELECT COUNT (DISTINCT bname) AS total_cities  FROM deposit;
```

5. Update the value dept_no to 10 where second character of emp. name is ‘m’.

```
UPDATE Employee SET dept_no = 10 WHERE emp_name LIKE ‘_m%’;
```

6. Update the value of employee name whose employee number is 103.

```
UPDATE Employee SET emp_name = ‘Adam’ WHERE emp_no = 103;
```

7. Write a query to display the current date. Label the column Date

```
SELECT GETDATE() AS Date;
```

8. For each employee, display the employee number, salary, and salary increased by 15% and expressed as a whole number. Label the column New Salary

SELECT emp_no , emp_sal , ROUND(emp_sal*1.15,0) AS ‘New Salary ‘ FROM Employee;

9. Modify your previous query to add a column that subtracts the old salary from the new salary. Label the column Increment.

SELECT emp_no , emp_sal, ROUND (emp_sal*1.150,0) AS ‘New Salary’, ROUND (emp_sal*0.15,0) AS Increment FROM Employee;

(ii) Write the SQL queries to perform numeric, date and String functions.

1. Retrieve all data from employee, jobs and deposit.

```
SELECT * FROM
employee; SELECT * FROM
jobs;
SELECT * FROM deposit;
```

2. Give details of account no. and deposited rupees of customers having account opened between dates 01-01-06 and 25-07-06.

```
SELECT a_no, amount FROM deposit
WHERE a_date BETWEEN '2006-01-01' AND '2006-07-25';
```

3. Display all jobs with minimum salary is greater than 4000.

```
 SELECT * FROM
jobs WHERE
min_sal > 4000
```

4. Display name and salary of employee whose department no is 20. Give alias name to name of employee.

```
SELECT emp_no, emp_name AS employee_name, emp_sal, dept_no
 FROM employee
 WHERE dept_no = 20;
```

5. Display employee no,name and department details of those employee whose department lies in(10,20)

```
SELECT emp_no, emp_name, dept_no FROM employee
 WHERE dept_no IN (10, 20);
```

6. Display all employee whose name start with ‘A’ and third character is ‘ ‘a’.

```
SELECT * FROM employee
WHERE emp_name LIKE 'A _a%';
```

7. Display name, number and salary of those employees whose name is 5 characters long and first three characters are ‘Ani’.

```
SELECT emp_name, emp_no, emp_sal FROM employee
WHERE emp_name LIKE 'Ani___';
```

8. Display the non-null values of employees and also employee name second character should be ‘n’ and string should be 5 character long.

```
SELECT * FROM employee
WHERE emp_name LIKE '_n%' AND LENGTH(emp_name) = 5 AND
emp_name IS NOT NULL;
```

9. Display the null values of employee and also employee name’s third character should be ‘a’.

```
SELECT * FROM employee
WHERE emp_name LIKE '__a%' AND emp_name IS NULL;
```

‘ HTML , CSS, JAVASCRIPT ’

Practical-18

Aim : Make a Resume using the HTML tags without CSS.

Code :

```
<!DOCTYPE html>
<html>
 <head>
  <title>Resume</title>
 </head>
 <body>
  <h1>Shreya Raval</h1>
  <p>Email: ravalshreya.2004@gmail.com</p>
  <p>Phone:91xxxxxxxx </p>
  <h2>Objective</h2>
  <p>Highly passionate and hard-working student seeking for internship in an esteemed organization. To enhance my technical skills and to explore more about innovative and inspiring field of Web Development.</p>
  <h2>Education</h2>
  <ul>
   <li>Bachelor of Technology in Computer Science, Silver Oak University, (pursuing)</li>
  </ul>
  <h2>Skills</h2>
  <ul>
   <li>HTML</li>
   <li>CSS</li>
   <li>C</li>
   <li>Adaptability</li>
  <li>Management Skills</li>
  </ul>
<h2>Hobbies</h2>
```

```
<ul>
<li>Reading</li>
<li>Listening Music</li>
</ul>
<h2>Declaration</h2>
<p>I hereby declare that the details furnished above are true and accurate to the best of my knowledge.</p>
</h2>
  </body>
</html>
```

Output :

Shreya Raval

Email: ravalshreya.2004@gmail.com

Phone:91xxxxxxxx

Objective

Highly passionate and hard-working student seeking for internship in an esteemed organization. To enhance my technical skills and to explore more about innovative and inspiring field of Web Development.

Education

- Bachelor of Technology in Computer Science, Silver Oak University, (pursuing)

Skills

- HTML
- CSS
- C
- Adaptability
- Management Skills

Hobbies

- Reading
- Listening Music

Declaration

I hereby declare that the details furnished above are true and accurate to the best of my knowledge.

Practical-19

Aim : Create an HTML webpage that shows Poster Presentation using all Table Properties.

Code :

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="UTF-8">
 <title>Poster Presentation</title>
 <style>
  table {
   border-collapse: collapse;
   width: 100%;
   max-width: 800px;
   margin: 0 auto;
  }

  th, td {
   padding: 10px;
   border: 1px solid #ccc;
  }

  th {
   background-color: #f2f2f2;
  }

  .poster-title {
   font-size: 24px;
   font-weight: bold;
   text-align: center;
   margin-bottom: 30px;
  }

  .poster-author {
```

```
    font-size: 18px;
    font-weight: bold;
    margin-bottom: 20px;
   }

   .poster-body {
    font-size: 16px;
    line-height: 1.5;
    margin-bottom: 40px;
   }

   .poster-image {
    width: 100%;
    max-width: 600px;
    display: block;
    margin: 0 auto;
    margin-bottom: 30px;
   }

   .poster-table {
    margin-bottom: 50px;
   }

   .poster-footer {
    font-size: 14px;
    text-align: center;
   }
  </style>
</head>
<body>
  <div class="poster-title">Title of Poster Presentation Goes Here</div>

  <div class="poster-author">Author Name(s)</div>

  <div class="poster-body">
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce eget arcu vel quam fringilla rhoncus. Duis sit amet ultrices ante. Nullam auctor ex vel sapien feugiat, vitae sollicitudin lacus vestibulum. Nullam malesuada, justo et malesuada auctor, ex ante sagittis dolor, at congue nibh felis non libero. In hac habitasse platea dictumst. Integer interdum sapien ut neque sagittis lacinia. Nulla vel ipsum sed sapien malesuada sagittis. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nullam laoreet lorem a sapien aliquam dictum. Fusce id ex vitae mi luctus dictum.</p>
    <p>Donec a ligula ut velit tincidunt fringilla a non ipsum. In volutpat mauris in arcu pretium facilisis. Pellentesque habitant morbi tristique senectus et netus et malesuada
```

fames ac turpis egestas. Etiam ut quam et nunc blandit mollis. In non mauris vitae ante elementum ut a arcu. Aliquam gravida dolor ut urna malesuada blandit. Aliquam vel metus et felis elementum auctor. Donec euismod sapien vel nibh facilisis, vel malesuada lectus vestibulum. Duis feugiat, nibh nec euismod imperdiet, lectus turpis consequat magna, vel ultrices quam mauris ac velit. Proin sodales turpis id suscipit pellentesque. In eleifend nisi sed leo dictum, vel tincidunt odio congue.

Sed tempor hendrerit eros, eu pulvinar augue consectetur eu. Etiam imperdiet libero quis velit vestibulum blandit.

Output :

Title of Poster Presentation Goes Here

Author Name(s)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce eget arcu vel quam fringilla rhoncus. Duis sit amet ultrices ante. Nullam auctor ex vel sapien feugiat, vitae sollicitudin lacus vestibulum. Nullam malesuada, justo et malesuada auctor, ex ante sagittis dolor, at congue nibh felis non libero. In hac habitasse platea dictumst. Integer interdum sapien ut neque sagittis lacinia. Nulla vel ipsum sed sapien malesuada sagittis. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nullam laoreet lorem a sapien aliquam dictum. Fusce id ex vitae mi luctus dictum.

Donec a ligula ut velit tincidunt fringilla a non ipsum. In volutpat mauris in arcu pretium facilisis. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Etiam ut quam et nunc blandit mollis. In non mauris vitae ante elementum ut a arcu. Aliquam gravida dolor ut urna malesuada blandit. Aliquam vel metus et felis elementum auctor. Donec euismod sapien vel nibh facilisis, vel malesuada lectus vestibulum. Duis feugiat, nibh nec euismod imperdiet, lectus turpis consequat magna, vel ultrices quam mauris ac velit. Proin sodales turpis id suscipit pellentesque. In eleifend nisi sed leo dictum, vel tincidunt odio congue.

Sed tempor hendrerit eros, eu pulvinar augue consectetur eu. Etiam imperdiet libero quis velit vestibulum blandit.

Practical-20

Aim : Create an HTML page table and form.

Code :

```
<!DOCTYPE html>
<html>
<head>
 <meta charset="UTF-8">
 <title>Table and Form Example</title>
</head>
<body>
 <h1>Table Example</h1>
 <table>
  <caption>Employee Information</caption>
  <thead>
   <tr>
    <th>Employee ID</th>
    <th>Name</th>
    <th>Department</th>
    <th>Salary</th>
   </tr>
  </thead>
  <tbody>
   <tr>
    <td>101</td>
    <td>John Doe</td>
    <td>Marketing</td>
    <td>$50,000</td>
   </tr>
   <tr>
    <td>102</td>
    <td>Jane Smith</td>
    <td>IT</td>
    <td>$70,000</td>
   </tr>
   <tr>
    <td>103</td>
    <td>Bob Johnson</td>
    <td>Finance</td>
    <td>$90,000</td>
   </tr>
  </tbody>
 </table>

 <h1>Form Example</h1>
```

```
 <form>
   <label for="name">Name:</label>
   <input type="text" id="name" name="name"><br><br>

   <label for="email">Email:</label>
   <input type="email" id="email" name="email"><br><br>

   <label for="password">Password:</label>
   <input type="password" id="password" name="password"><br><br>

   <label for="comment">Comment:</label><br>
   <textarea id="comment" name="comment" rows="5" cols="40"></textarea><br><br>

   <input type="submit" value="Submit">
 </form>
</body>
</html>
```

Output :

Table Example

Employee Information

Employee ID	Name	Department	Salary
101	John Doe	Marketing	$50,000
102	Jane Smith	IT	$70,000
103	Bob Johnson	Finance	$90,000

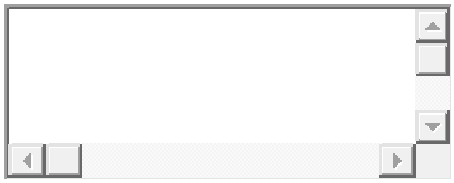
Form Example

Name:

Email:

Password:

Comment:



Submit

Practical-21

Aim : Create Registration form and do proper validation with HTML 5 inbuilt functionality.

Code :

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Registration Form</title>
</head>
<body>
  <h1>Registration Form</h1>
  <form method="POST" action="submit-registration.php">
    <label for="fullname">Full Name:</label>
    <input type="text" id="fullname" name="fullname" required><br><br>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required><br><br>

    <label for="password">Password:</label>
    <input type="password" id="password" name="password" minlength="8"
required><br><br>

    <label for="confirm-password">Confirm Password:</label>
    <input type="password" id="confirm-password" name="confirm-password"
minlength="8" required
      oninput="checkPasswordMatch()"><br>
    <small id="password-match-error" style="color: red; display: none;">Passwords do not
match</small><br><br>

    <label for="age">Age:</label>
    <input type="number" id="age" name="age" min="18" max="99" required><br><br>

    <label for="gender">Gender:</label>
    <input type="radio" id="male" name="gender" value="male" required>
    <label for="male">Male</label>
    <input type="radio" id="female" name="gender" value="female">
    <label for="female">Female</label><br><br>

    <input type="submit" value="Submit">
```

```
 </form>

 <script>
  function checkPasswordMatch() {
    var password = document.getElementById("password");
    var confirm_password = document.getElementById("confirm-password");
    var error_message = document.getElementById("password-match-error");
    if (password.value != confirm_password.value) {
      error_message.style.display = "block";
    } else {
      error_message.style.display = "none";
    }
  }
 </script>
</body>
</html>
```

Output :

Registration Form

Full Name:

Email:

Password:

Confirm Password:

Age:

Gender: Male Female

Submit

Practical-22

Aim : Make a Resume using the HTML tags with CSS

Code :

```
<!DOCTYPE html>
<html>
```

```
<head>
	<meta charset="UTF-8">
	<title>Resume</title>
	<style>
		body {
			font-family: Arial, sans-serif;
			margin: 0;
			padding: 0;
		}

		header {
			background-color: #333;
			color: #fff;
			padding: 20px;
			text-align: center;
		}

		h1 {
			font-size: 36px;
			margin: 0;
		}

		h2 {
			font-size: 24px;
			margin-bottom: 5px;
		}

		section {
			padding: 20px;
			margin: 20px;
			border: 1px solid #ddd;
		}

		ul {
			margin: 0;
			padding: 0;
			list-style: none;
		}

		li {
			margin-bottom: 10px;
		}

		.contact-info {
```

```
                display: flex;
                align-items: center;
                margin-bottom: 10px;
            }

            .contact-info img {
                margin-right: 10px;
            }

            .skills {
                display: flex;
                flex-wrap: wrap;
                margin: 0;
                padding: 0;
            }

            .skills li {
                margin-right: 10px;
            }
        </style>
</head>
<body>
        <header>
            <h1>John Doe</h1>
            <h2>Web Developer</h2>
        </header>

        <section>
            <h2>Contact Information</h2>
            <ul>
                <li class="contact-info"><img src="phone-icon.png" alt="Phone
Icon"><span>(123) 456-7890</span></li>
                <li class="contact-info"><img src="email-icon.png" alt="Email
Icon"><span>john.doe@example.com</span></li>
                <li class="contact-info"><img src="address-icon.png" alt="Address
Icon"><span>1234 Main St, Anytown USA 12345</span></li>
            </ul>
        </section>

        <section>
            <h2>Skills</h2>
            <ul class="skills">
                <li>HTML</li>
                <li>CSS</li>
```

```
                <li>JavaScript</li>
                <li>PHP</li>
                <li>MySQL</li>
                <li>WordPress</li>
                <li>Responsive Web Design</li>
            </ul>
        </section>

        <section>
            <h2>Experience</h2>
            <h3>Web Developer, ABC Company</h3>
            <p><em>June 2019 - Present</em></p>
            <ul>
                <li>Developed and maintained company website using
WordPress</li>
                <li>Implemented responsive design for optimal viewing on mobile
devices</li>
                <li>Collaborated with marketing team to improve website traffic and
user engagement</li>
            </ul>
        </section>

        <section>
            <h2>Education</h2>
            <h3>Bachelor's Degree in Computer Science, XYZ University</h3>
            <p><em>Graduated May 2019</em></p>
            <ul>
                <li>Coursework included web development, database management,
and programming fundamentals</li>
                <li>Participated in hackathons and coding competitions, placing in the
top 10 in several
```

Output :

John Doe

Web Developer

Contact Information

- (123) 456-7890
- john.doe@example.com

- 1234 Main St, Anytown USA 12345

Skills

- HTML
- CSS
- JavaScript
- PHP
- MySQL
- WordPress
- Responsive Web Design

Experience

Web Developer, ABC Company

June 2019 - Present

- Developed and maintained company website using WordPress
- Implemented responsive design for optimal viewing on mobile devices
- Collaborated with marketing team to improve website traffic and user engagement

Education

Bachelor's Degree in Computer Science, XYZ University

Graduated May 2019

- Coursework included web development, database management, and programming fundamentals
- Participated in hackathons and coding competitions, placing in the top 10 in several

Practical-23

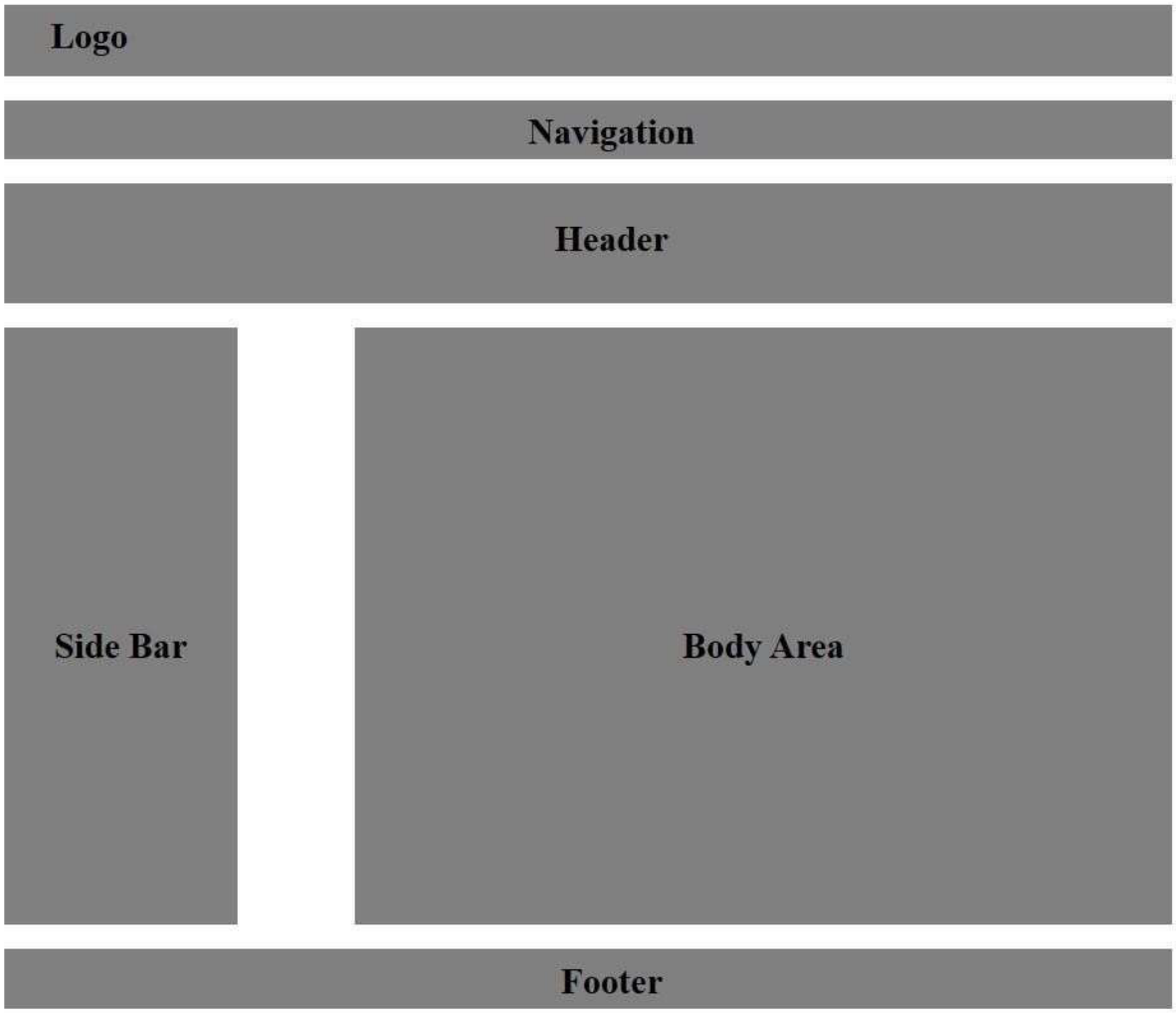
Aim : Create an HTML Page containing Gray Layout using CSS.

Code : <!DOCTYP
E html>

```
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Gray
layout 1</title>
<style>
*{font-size: 30px;font-weight: bolder;}
#r1{background-color: gray; height: 50px; width: 48%; padding-top: 10px;padding-left:
2%;margin-bottom: 20px;}
#r2,#r5{background-color: gray; height: 40px; width: 48%; padding-top:
10px;paddingleft:
2%;margin-bottom: 20px; text-align: center;}
#r3{background-color: gray; height: 70px; width: 48%; padding-top: 30px;padding-left:
2%;margin-bottom: 20px; text-align: center;}
#r4{height: 500px; width: 100%; margin-bottom: 20px;}
#r4 div{float: left; background-color: gray; padding-top: 250px; height: 250px; textalign:
center;}
#r4c1{width: 10%; margin-right: 5%; }
#r4c2{width: 35%; }
</style>
</head>
<body>
<div id="maindiv">
<div id="r1">Logo</div>
<div id="r2">Navigation</div>
<div id="r3">Header</div>
<div id="r4">
<div id="r4c1">Side Bar</div>
<div id="r4c2">Body Area</div>
</div>
<div id="r5">Footer</div>
</div>
</body>
</html>
```

Output :

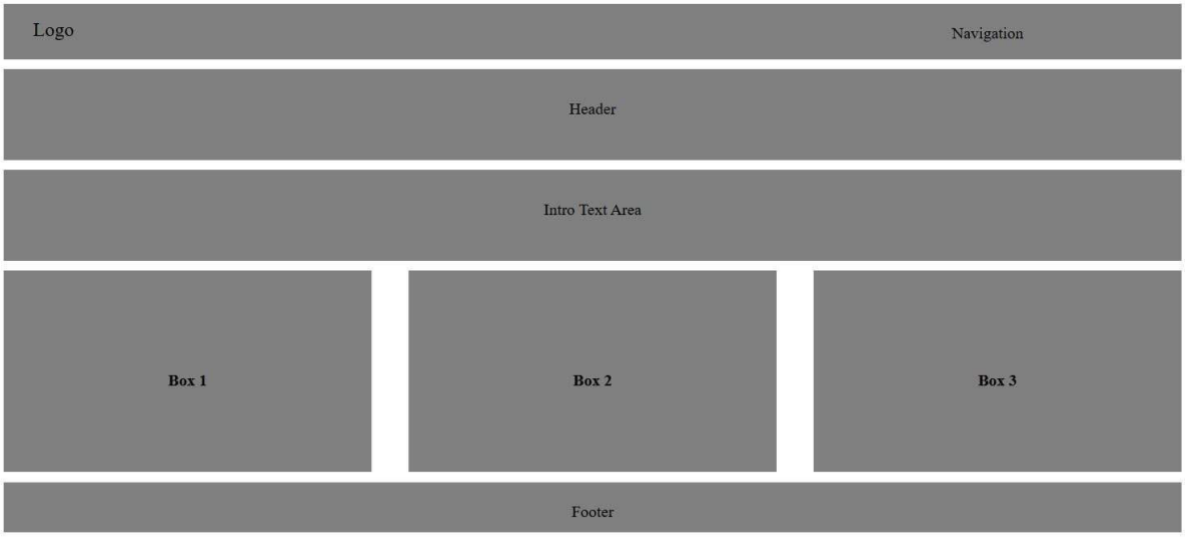


Code : <!DOCTYP

```
E html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Gray
Layout 2</title>
<style>
#r1{background-color: gray; width: 78%;padding-top: 15px;padding-left: 2%; marginleft:
10%; height: 40px;}
#r1c1{float: left; font-size: larger;}
#r1c2{margin-left:80%;padding-top: 5px;}
#r2, #r3{width: 80%;margin-left: 10%;height: 60px;padding-top: 30px; backgroundcolor:
gray;margin-top: 10px; text-align: center;}
#r4 div{float: left; background-color: gray; font-weight: bolder;}
#r4c1{margin-left: 10%;text-align: center;width: 25%;height: 100px;margin-top:
10px;paddingtop: 100px;}
#r4c2{margin-left: 2.5%;text-align: center;width: 25%;height: 100px;margin-top:
10px;paddingtop: 100px;}
#r4c3{margin-left: 2.5%;text-align: center;width: 25%;height: 100px;margin-top:
```

```
10px;paddingtop: 100px;}
#r5{width: 80%;margin-left: 10%;background-color: graY;margin-top: 220px;height:
30px;padding-top: 20px;text-align: center;}
</style>
</head>
<body>
<div id="r1">
<div id="r1c1">Logo</div>
<div id="r1c2">Navigation</div>
</div>
<div id="r2">Header</div>
<div id="r3">Intro Text Area</div>
<div id="r4">
<div id="r4c1">Box 1</div>
<div id="r4c2">Box 2</div>
<div id="r4c3">Box 3</div>
</div>
<div id="r5">Footer</div>
</body>
</html>
```

Output :



Code :

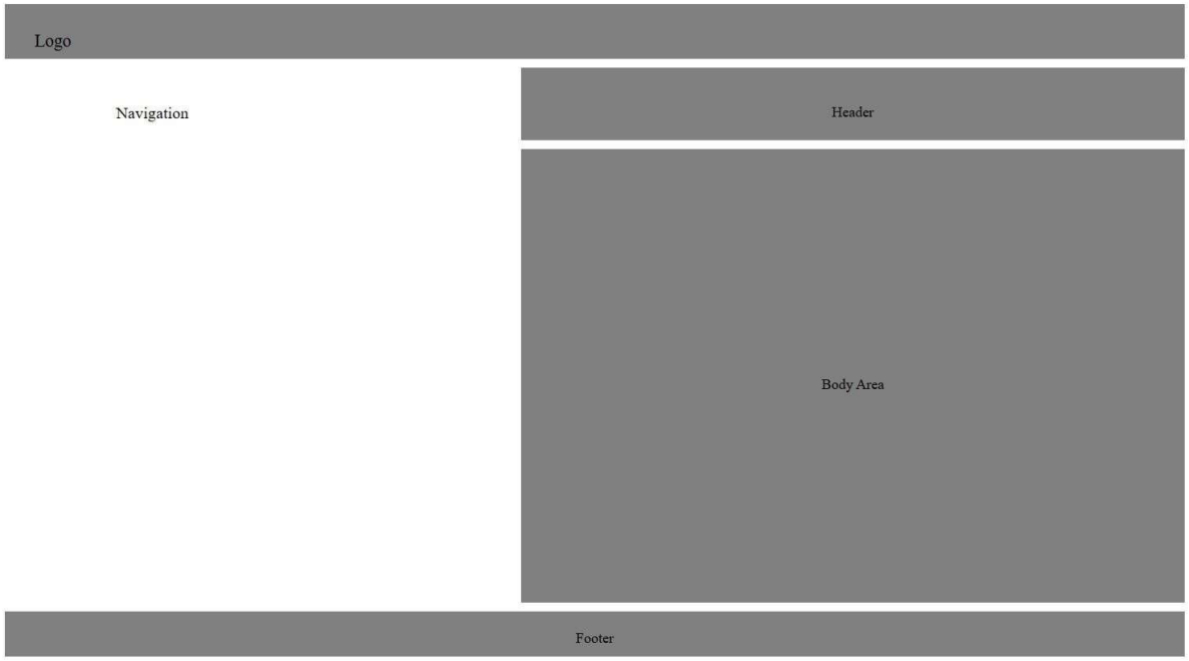
```
<!DOCTYP
E html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Gray
Layout 3</title>
<style>
#r1{padding-top: 30px;height: 30px; margin-top: 10px;margin-left: 10%;width:
78%;paddingleft: 2%; background-color: gray;font-size: larger;}
```

#r2 div{float: left;margin-left: 10%;margin-top: 10px;height: 40px;padding-top: 40px;
text-align: center;}
#r2c1{ width: 20%;margin-right: 5%;font-size: large;}
#r2c2{width: 45%; background-color: gray;}
#r3{margin-bottom: 10px; margin-left: 45%;width: 45%; height: 250px;padding-top:
250px;textalign: center;background-color: gray;margin-top: 100px;}
#r4{margin-left: 10%; width: 80%;text-align: center;padding-top: 20px;height:
30px;backgroundcolor: gray;} </style>
</head>
<body>
Logo

Navigation
Header

Body Area
Footer
</body>
</html>

Output :



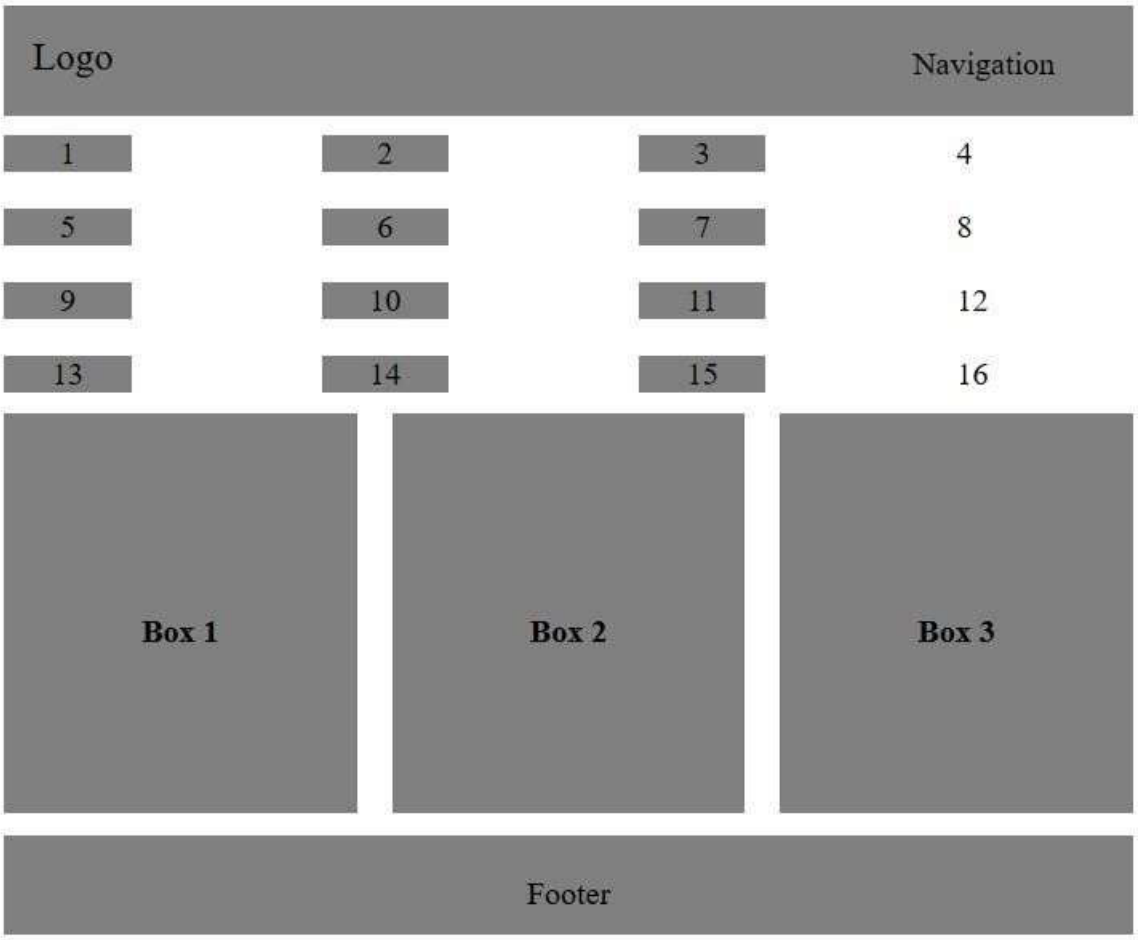
Code :

<!DOCTYP
E html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Gray
Layout 4</title>
<style>
#r1{background-color: gray; width: 78%;padding-top: 15px;padding-left: 2%; marginleft:
10%; height: 40px;}
#r1c1{float: left; font-size: larger;}

```
#r1c2{margin-left:80%;padding-top: 5px;}
#r3{margin-bottom: 10px;height: 100px; margin-left: 10%;margin-top: 10px;}
#c1,#c2,#c3{float: left; background-color: gray; width: 10%;margin-right: 15%; textalign:
center;}
#c5,#c6,#c7{float: left;background-color: gray; width: 10%; margin-right: 15%;text-align:
center;} #c9,#c10,#c11{float: left;background-color: gray; width: 10%; margin-right:
15%;text-align:
center;}
#c13,#c14,#c15{float: left;background-color: gray; width: 10%; margin-right:
15%;textalign: center;}
#r4 div{float: left; background-color: gray; font-weight: bolder;}
#r4c1{margin-left: 10%;text-align: center;width: 25%;height: 100px;margin-top:
10px;paddingtop: 100px;}
#r4c2{margin-left: 2.5%;text-align: center;width: 25%;height: 100px;margin-top:
10px;paddingtop: 100px;}
#r4c3{margin-left: 2.5%;text-align: center;width: 25%;height: 100px;margin-top:
10px;paddingtop: 100px;}
#r5{width: 80%;margin-left: 10%;background-color: graY;margin-top: 250px;height:
30px;padding-top: 20px;text-align: center;}
</style>
</head>
<body>
<div id="r1">
<div id="r1c1">Logo</div>
<div id="r1c2">Navigation</div>
</div>
<div>
<div></div>
</div>
<div id="r3">
<div id="c1">1</div>
<div id="c2">2</div>
<div id="c3">3</div>
<div id="c4">4</div><br>
<div id="c5">5</div>
<div id="c6">6</div>
<div id="c7">7</div>
<div id="c8">8</div><br>
<div id="c9">9</div>
<div id="c10">10</div>
<div id="c11">11</div>
<div id="c12">12</div><br>
<div id="c13">13</div>
<div id="c14">14</div>
<div id="c15">15</div>
<div id="c16">16</div>
</div>
<div id="r4">
<div id="r4c1">Box 1</div>
<div id="r4c2">Box 2</div>
<div id="r4c3">Box 3</div>
</div>
<div id="r5">Footer</div>
</body>
```

</html>

Output :



Practical-24

Aim : Demonstrate JavaScript Form Validation with proper examples.

Code :

Validate a required input field

In this example, we want to ensure that a user has entered a value in a required input field before submitting the form. We can use JavaScript to check whether the field is empty, and display an error message if it is.

```
<form>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required>
  <button type="submit">Submit</button>
</form>

<script>
  const form = document.querySelector('form');
```

```
  const nameInput = document.querySelector('#name');

  form.addEventListener('submit', (event) => {
    if (!nameInput.value) {
      event.preventDefault();
      alert('Please enter your name');
    }
  });
</script>
```

In this example, we use the `required` attribute on the input field to ensure that the user enters a value before submitting the form. We also use JavaScript to listen for the `submit` event on the form. When the user submits the form, we check whether the `value` property of the `nameInput` element is empty. If it is, we prevent the default form submission using the `preventDefault()` method on the event object, and display an alert message asking the user to enter their name

Validate a password field

In this example, we want to ensure that a user has entered a valid password before submitting the form. We can use JavaScript to check whether the password meets certain requirements, such as containing at least one uppercase letter, one lowercase letter, and one digit.

```
<form>
  <label for="password">Password:</label>
  <input type="password" id="password" name="password">
  <button type="submit">Submit</button>
</form>


<script>
  const form = document.querySelector('form');
  const passwordInput = document.querySelector('#password');
```

```
  form.addEventListener('submit', (event) => {

    const password = passwordInput.value;



    if (password.length < 8 || !/[A-Z]/.test(password) || !/[a-z]/.test(password) ||
!/\d/.test(password)) {

      event.preventDefault();

      alert('Your password must be at least 8 characters long and contain at least one
uppercase letter, one lowercase letter, and one digit.');

    }

  });

</script>
```

In this example, we use a regular expression to check whether the password meets certain requirements. The regular expression checks whether the password is at least 8 characters long (`password.length < 8`), and contains at least one uppercase letter (`/[A-Z]/.test(password)`), one lowercase letter (`/[a-z]/.test(password)`), and one digit (`/\d/.test(password)`). If any of these conditions is not met, we prevent the default form submission using the `preventDefault()` method on the event object, and display an alert message asking the user to enter a valid password.

Validate an email field

In this example, we want to ensure that a user has entered a valid email address before submitting the form. We can use JavaScript to check whether the email address is formatted correctly, using a regular expression.

```
<form>

  <label for="email">Email:</label>

  <input type="email" id="email" name="email">

  <button type="submit">Submit</button>
```

```
</form>



<script>

 const form = document.querySelector('form');

 const emailInput = document.querySelector('#email');



 form.addEventListener('submit', (event) => {

   const email
```

Practical-25

Aim : Write a JavaScript to check if the number is even or odd.

Code :

```
function checkEvenOrOdd(number) {
  if (number % 2 === 0) {
    return "Even";
  } else {
    return "Odd";
  }
}
```

Output :

In this example, we define a function called `checkEvenOrOdd` that takes one parameter, `number`. We then use the modulus operator (`%`) to check if `number` is divisible by 2. If the remainder is 0, we return the string "Even". If the remainder is not 0, we return the string "Odd".

You can call this function with any number to check whether it's even or odd. For example:

```
console.log(checkEvenOrOdd(2)); // outputs "Even"

console.log(checkEvenOrOdd(3)); // outputs "Odd"

console.log(checkEvenOrOdd(42)); // outputs "Even"
```

```
console.log(checkEvenOrOdd(99)); // outputs "Odd"
```

Practical-26

Aim : Create a page and access the LocationAPI.

Code :

```
<!DOCTYPE html>
<html>
  <head>
    <title>LocationAPI Example</title>
  </head>
  <body>
    <h1>LocationAPI Example</h1>
    <p>Latitude: <span id="latitude"></span></p>
    <p>Longitude: <span id="longitude"></span></p>

    <script>
      function getLocation() {
        if (navigator.geolocation) {
          navigator.geolocation.getCurrentPosition(showPosition);
        } else {
          alert("Geolocation is not supported by this browser.");
        }
      }

      function showPosition(position) {
        document.getElementById("latitude").innerHTML = position.coords.latitude;
        document.getElementById("longitude").innerHTML = position.coords.longitude;
      }

      getLocation();
    </script>
  </body>
</html>
```

Output :

When you load this HTML page in a browser that supports the LocationAPI and has location services enabled, you should see something like this:
LocationAPI Example

Latitude: 37.7749
Longitude: -122.4194

Practical-27

Aim : Create a simple XMLHTTPRequest,and retrieve the data from the text file.

Code :

```
        // create a new XMLHttpRequest object
var xhttp = new XMLHttpRequest();

// specify the HTTP method and URL of the file to retrieve
xhttp.open("GET", "example.txt", true);

// set a callback function to handle the response
xhttp.onreadystatechange = function() {
  if (this.readyState == 4 && this.status == 200) {
    // handle the response here
    console.log(this.responseText);
  }
};

// send the request
xhttp.send();
```

Output (Explanation) :

In this example, we create a new XMLHttpRequest object using the `new XMLHttpRequest()` constructor. We then use the `open()` method to specify the HTTP method (in this case, "GET") and the URL of the file we want to retrieve. The third parameter of the `open()` method is a boolean that specifies whether the request should be asynchronous or not; we set it to `true` so that the request will be sent asynchronously.

Next, we set a callback function using the `onreadystatechange` event handler. This function will be called whenever the `readyState` property of the `XMLHttpRequest` object changes. In this case, we check if the `readyState` is `4` (indicating that the request is complete) and if the `status` is `200` (indicating that the request was successful). If both

conditions are met, we handle the response by logging the response text to the console.

Finally, we send the request using the `send()` method. When the response is received, the callback function will be called automatically.