Lab Assignment 4

Ques1:
Write a Program to convert the binary number into a decimal number.

Ans:
```
DELIMITER $$
CREATE PROCEDURE BinaryToDecimal(IN binary_num VARCHAR(32))
BEGIN
   DECLARE decimal_num INT DEFAULT 0;
   DECLARE i INT DEFAULT 1;
   DECLARE rem INT;
   DECLARE len INT;
   SET len = LENGTH(binary_num);
   WHILE len > 0 DO
      SET rem = SUBSTRING(binary_num, len, 1);
      SET decimal_num = decimal_num + rem * i;
      SET i = i * 2;
      SET len = len - 1;
   END WHILE;
   SELECT decimal_num AS 'Decimal Value';
END$$
DELIMITER ;
CALL BinaryToDecimal('1010');
```

	Decimal Value
▶	10

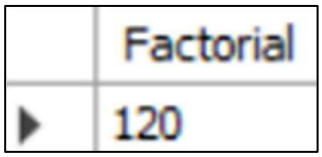
Ques2:
Write a Program to check if the year is a leap year or not.

Ans:
```
DELIMITER $$
CREATE PROCEDURE CheckLeapYear(IN year INT)
BEGIN
   IF (year % 4 = 0 AND year % 100 != 0) OR (year % 400 = 0) THEN
      SELECT CONCAT(year, ' is a leap year') AS Result;
   ELSE
      SELECT CONCAT(year, ' is not a leap year') AS Result;
   END IF;
END$$
DELIMITER ;
CALL CheckLeapYear(2024);
```

	Result
▶	2024 is a leap year

Ques3:
Write a program to Factorial of a Number.

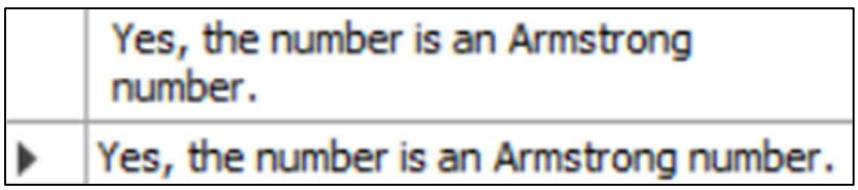
Ans:
```
DELIMITER $$
CREATE PROCEDURE Factorial(IN num INT)
BEGIN
   DECLARE fact INT DEFAULT 1;
   DECLARE i INT DEFAULT 1;
   WHILE i <= num DO
      SET fact = fact * i;
      SET i = i + 1;
   END WHILE;
   SELECT fact AS 'Factorial';
END$$
DELIMITER ;
CALL Factorial(5);
```



	Factorial
▶	120

Ques4:
Write a Program to Check if a number is an Armstrong number or not.

Ans:
```
DELIMITER //
CREATE PROCEDURE is_armstrong_number(IN number INT)
BEGIN
   DECLARE sum INT;
   DECLARE temp INT;
   DECLARE digits INT;
   SET sum = 0;
   SET temp = number;
   SET digits = LENGTH(CAST(number AS CHAR));
   WHILE temp > 0 DO
      SET sum = sum + POWER(temp MOD 10, digits);
      SET temp = FLOOR(temp / 10);
   END WHILE;
   IF sum = number THEN
      SELECT 'Yes, the number is an Armstrong number.';
   ELSE
      SELECT 'No, the number is not an Armstrong number.';
   END IF;
END //
DELIMITER ;
CALL is_armstrong_number(153);
```



	Yes, the number is an Armstrong number.
▶	Yes, the number is an Armstrong number.

Ques5:
Write a program to Find all the roots of a quadratic equation.

Ans:
```
DELIMITER $$
CREATE PROCEDURE QuadraticRoots(IN a DOUBLE, IN b DOUBLE, IN c DOUBLE)
BEGIN
   DECLARE discriminant DOUBLE;
   DECLARE root1 DOUBLE;
   DECLARE root2 DOUBLE;
   SET discriminant = b*b - 4*a*c;
   IF discriminant > 0 THEN
      SET root1 = (-b + SQRT(discriminant)) / (2 * a);
      SET root2 = (-b - SQRT(discriminant)) / (2 * a);
      SELECT root1 AS 'Root 1', root2 AS 'Root 2';
   ELSEIF discriminant = 0 THEN
      SET root1 = -b / (2 * a);
      SELECT root1 AS 'Root';
   ELSE
      SELECT 'Complex roots' AS Result;
   END IF;
END$$
DELIMITER ;
CALL QuadraticRoots(1, -7, 12);
```

	Root 1	Root 2
▶	4	3

Ques6:
Check whether a number is a palindrome.

Ans:
```
DELIMITER $$
CREATE PROCEDURE CheckPalindrome(IN num INT)
BEGIN
   DECLARE reverse_num INT DEFAULT 0;
   DECLARE temp INT DEFAULT num;
   DECLARE remainder INT;
   WHILE temp > 0 DO
      SET remainder = temp % 10;
      SET reverse_num = reverse_num * 10 + remainder;
      SET temp = FLOOR(temp / 10);
   END WHILE;
   IF reverse_num = num THEN
      SELECT CONCAT(num, ' is a palindrome') AS Result;
   ELSE
      SELECT CONCAT(num, ' is not a palindrome') AS Result;
   END IF;
END$$
DELIMITER ;
CALL CheckPalindrome(121);
```

	Result
▶	121 is a palindrome

Ques7:
Write a PL/SQL program to find the GCD of two numbers.

Ans:
```
DELIMITER $$
CREATE PROCEDURE GCD(IN num1 INT, IN num2 INT)
BEGIN
   DECLARE a INT;
   DECLARE b INT;
   DECLARE temp INT;
   SET a = num1;
   SET b = num2;
   WHILE b != 0 DO
      SET temp = b;
      SET b = a % b;
      SET a = temp;
   END WHILE;
   SELECT a AS 'GCD';
END$$
DELIMITER ;
CALL GCD(54, 24);
```

	GCD
▶	6

Ques8:
Write a Program to create a pyramid pattern.

Ans:
```
SELECT REPEAT(' ', 5 - level) AS space, REPEAT('*', 2 * level - 1) AS stars
FROM (SELECT 1 AS level UNION SELECT 2 UNION SELECT 3 UNION SELECT 4 UNION
SELECT 5) AS levels;
```

space	stars
	*
	***
	*****
	*******
	*********

Ques9:
Write a program to show the use of trigger after the entry of data in a table.

Ans:

```
-- Drop tables if they exist (cleanup)
DROP TABLE IF EXISTS employees;
DROP TABLE IF EXISTS audit_log;

-- 1. Create the main table
CREATE TABLE employees (
    employee_id INT AUTO_INCREMENT PRIMARY KEY,
    employee_name VARCHAR(100),
    department VARCHAR(100),
    salary DECIMAL(10, 2)
);

-- 2. Create the audit log table
CREATE TABLE audit_log (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    operation VARCHAR(50),
    employee_id INT,
    log_time DATETIME
);

-- 3. Create the trigger to log INSERT operations
DELIMITER $$
CREATE TRIGGER after_employee_insert
AFTER INSERT ON employees
FOR EACH ROW
BEGIN
    INSERT INTO audit_log (operation, employee_id, log_time)
    VALUES ('INSERT', NEW.employee_id, NOW());
END$$
DELIMITER ;

-- 4. Insert data into the 'employees' table
INSERT INTO employees (employee_name, department, salary) VALUES ('John Doe', 'Engineering', 75000);

-- 5. Select from both tables to check the results
SELECT * FROM employees;
SELECT * FROM audit_log;
```

	employee_id	employee_name	department	salary
▶	1	John Doe	Engineering	75000.00
*	NULL	NULL	NULL	NULL

	log_id	operation	employee_id	log_time
▶	1	INSERT	1	2024-10-14 15:47:54
*	NULL	NULL	NULL	NULL

Ques10:
Write a program to show the change in data due to the update query on a table using trigger.

Ans:

```
-- 1. Create the trigger to log UPDATE operations
DELIMITER $$
CREATE TRIGGER after_employee_update
AFTER UPDATE ON employees
FOR EACH ROW
BEGIN
    INSERT INTO audit_log (operation, employee_id, log_time)
    VALUES ('UPDATE', NEW.employee_id, NOW());
END$$
DELIMITER;

-- 2. Update data in the 'employees' table
UPDATE employees SET salary = 80000 WHERE employee_id = 1;

-- 3. Select from both tables to check the results
SELECT * FROM employees;
SELECT * FROM audit_log;
```

	employee_id	employee_name	department	salary
▶	1	John Doe	Engineering	80000.00
*	NULL	NULL	NULL	NULL

	log_id	operation	employee_id	log_time
▶	1	INSERT	1	2024-10-14 15:47:54
	2	UPDATE	1	2024-10-14 15:49:21
*	NULL	NULL	NULL	NULL