**Batch -** T7

**Assignment No. -** 2

**Title –** MySQL / PSM Review and Object Relational Databases

**Student Name -** Sharaneshwar Bharat Punjal

**Student PRN -** 23520011

**I. MySQL / PSM Review :**

a) Create a table called test\_table with 2 columns RecordNumber (type : Number(3)) and CurrentDate (type : Date)). Write a procedure in PSM which will insert 50 records into test\_table. Insert the current date value into the table.

**1. Create table:**

create table test\_table(

RecordNumber INT(3),

CurrentDate Date

);

**2. Make a procedure to insert the 50 values in the table:**

DELIMITER $$

create procedure addvalues()

BEGIN

Declare i int default 1;

while i<= 50 do

insert into test\_table(RecordNumber,CurrentDate)

values (i,CURDATE());

set i = i+1;

end while;

END $$

DELIMITER ;

**3. Call the procedure:**

call addvalues();

**4. Get all the rows of inserted value:**

select \* from test\_table;

A screenshot of a computer

Description automatically generated

b) Create a products table products (ProductID number(4), category char(3), detail varchar2(30), price number(10,2), stock number(5)). Insert the sample data. Write PSM procedure with two arguments X & Y which will increase price by X% for all products in category Y. X and Y will be given by user.

**1. Create table:**

create table products(

ProductId int(4),

category varchar(3),

detail varchar(30),

price decimal(10,2),

stock int(5)

);

**2. Insert Values:**

INSERT INTO products (ProductId, category, detail, price, stock)

VALUES

(1001, 'ELE', 'Electric Kettle', 1999.99, 50),

(1002, 'FUR', 'Wooden Chair', 999.50, 100),

(1003, 'APP', 'Smartphone', 29999.00, 30),

(1004, 'KIT', 'Mixer Grinder', 2499.99, 75),

(1005, 'ELE', 'Microwave Oven', 7999.00, 40),

(1006, 'FAS', 'Leather Jacket', 5499.99, 20);

**3. Make a procedure:**

DELIMITER $$

create procedure increval(in x int , in y varchar(3))

begin

UPDATE products

SET price = price + (price \* x / 100)

WHERE category = y;

end $$

DELIMITER ;

**4. Call the procedure:**

CALL increval(0, 'ELE');

**5. Get all the rows of inserted value:**

A screenshot of a computer

Description automatically generated

**II. Object Relational Databases:**

a) Create Object Table containing field “name” of size 50 characters and member function “countNoOfWords” which returns the no. of words in “name” field. Demonstrate the working by entering different data.

**1. Create table:**

CREATE TABLE ObjectTable (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(50)

);

**2. Make a procedure:**

DELIMITER $$

CREATE FUNCTION countNoOfWords(nameField VARCHAR(50))

RETURNS INT

DETERMINISTIC

BEGIN

RETURN LENGTH(nameField) - LENGTH(REPLACE(nameField, ' ', '')) + 1;

END $$

DELIMITER ;

**3. Insert the values:**

INSERT INTO ObjectTable (name) VALUES

('Alice in Wonderland'),

('MySQL Workbench'),

('Database Design Basics'),

('Hello World');

**4. Get all the rows of inserted value:**

SELECT id, name, countNoOfWords(name) AS word\_count

FROM ObjectTable;.

**A screenshot of a computer

Description automatically generated**

b) Create an address type with the following attributes : address, city, state & pincode. Include the following methods i. to extract the addresses based on given keyword. 1.

**1. Create table:**

CREATE TABLE AddressTable (

id INT AUTO\_INCREMENT PRIMARY KEY,

address VARCHAR(255),

city VARCHAR(50),

state VARCHAR(50),

pincode VARCHAR(10)

);

**2. Make a procedure:**

DELIMITER $$

CREATE FUNCTION extractAddressesByKeyword(keyword VARCHAR(50))

RETURNS TEXT

DETERMINISTIC

BEGIN

DECLARE result TEXT;

SELECT GROUP\_CONCAT(CONCAT(address, ', ', city, ', ', state, ', ', pincode) SEPARATOR '; ')

INTO result

FROM AddressTable

WHERE address LIKE CONCAT('%', keyword, '%')

OR city LIKE CONCAT('%', keyword, '%')

OR state LIKE CONCAT('%', keyword, '%')

OR pincode LIKE CONCAT('%', keyword, '%');

RETURN IFNULL(result, 'No addresses found');

END $$

DELIMITER ;

**3. Another procedure:**

DELIMITER $$

CREATE FUNCTION countWordsInField(fieldName VARCHAR(50), idValue INT)

RETURNS INT

DETERMINISTIC

BEGIN

DECLARE fieldContent VARCHAR(255);

DECLARE wordCount INT;

SET fieldContent = CASE

WHEN fieldName = 'address' THEN (SELECT address FROM AddressTable WHERE id = idValue)

WHEN fieldName = 'city' THEN (SELECT city FROM AddressTable WHERE id = idValue)

WHEN fieldName = 'state' THEN (SELECT state FROM AddressTable WHERE id = idValue)

WHEN fieldName = 'pincode' THEN (SELECT pincode FROM AddressTable WHERE id = idValue)

ELSE NULL

END;

SET wordCount = LENGTH(fieldContent) - LENGTH(REPLACE(fieldContent, ' ', '')) + 1;

RETURN IFNULL(wordCount, 0);

END $$

DELIMITER ;

**4. Insert Values:**

INSERT INTO AddressTable (address, city, state, pincode) VALUES

('123 Main Street', 'New York', 'New York', '10001'),

('456 Elm Street', 'Los Angeles', 'California', '90001'),

('789 Oak Avenue', 'Chicago', 'Illinois', '60601'),

('101 Pine Lane', 'Houston', 'Texas', '77001');

**5. Result:**

SELECT extractAddressesByKeyword('Street') AS result;

A screenshot of a computer

Description automatically generated

SELECT countWordsInField('address', 1) AS word\_count;j. to return the no. of words in each given field (method should accept the name of attribute/field)

A screenshot of a computer

Description automatically generated

c) Create a user defined data type course\_Type with 2 attributes course\_id, description:

i. Create an object table based on the type created.

j. Insert rows into the table

**1. Create table:**

CREATE TYPE course\_Type AS (

course\_id INT,

description TEXT

);

**2. Create Table of type course\_type:**

CREATE TABLE course\_table OF course\_Type;

**3. Insert the values:**

INSERT INTO course\_table VALUES (101, 'Introduction to Programming');

INSERT INTO course\_table VALUES (102, 'Data Structures and Algorithms');

INSERT INTO course\_table VALUES (103, 'Database Management Systems');

INSERT INTO course\_table VALUES (104, 'Artificial Intelligence');

**4. Get all the rows of inserted value:**

SELECT \* FROM course\_table;

A screenshot of a computer

Description automatically generated