**Assignment 1: SQL Assignment**

**Question 1:**

Table Name : Employee

|  |  |  |
| --- | --- | --- |
| EID | Ename | Pid |
| 100 | ABC | P189 |
| 101 | MNP | P789 |
| 102 | DEF | P567 |
| 103 | JKL | NULL |
| 104 | GHJ | P567 |
| 105 | UJM | P189 |
| 106 | UJM | NULL |
| 107 | RGJ | P567 |
| 108 | IJH | NULL |
| 109 | NMH | P547 |
| 110 | SDF | P189 |
| 111 | TGB | P546 |
| 112 | HUF | NULL |
| 113 | SDF | P789 |
| 114 | RGJ | P567 |
| 115 | JKG | P189 |

Table Name : Project

|  |  |
| --- | --- |
| Pid | Pname |
| P189 | Rest |
| P546 | Spring |
| P547 | Maven |
| P567 | Swift |
| P789 | Angular |
| P799 | Script |
| P805 | Mssql |
| P855 | Perl |
| P895 | Python |
| P902 | Ruby |
| P928 | R |

**--Creating Tables**

CREATE TABLE Project (

Pid VARCHAR(54) NOT NULL,

Pname VARCHAR(50) NOT NULL,

PRIMARY KEY (Pid)

);

CREATE TABLE Employee (

Eid INT NOT NULL,

Ename VARCHAR(50) NOT NULL,

Pid VARCHAR(4),

PRIMARY KEY (Eid),

FOREIGN KEY (Pid) REFERENCES Project(Pid)

);

**--Inserting Values in Tables**

INSERT INTO Project (Pid, Pname)

VALUES ('P189', 'Rest'),

('P546', 'Spring'),

('P547', 'Maven'),

('P567', 'Swift'),

('P789', 'Angular'),

('P799', 'Script'),

('P805', 'Mssql'),

('P855', 'Perl'),

('P895', 'Pthon'),

('P902', 'Ruby'),

('P928', 'R');

INSERT INTO Employee (Eid, Ename, Pid)

VALUES (100, 'ABC', 'P189'),

(101, 'MNP', 'P789'),

(102, 'DEF', 'P567'),

(103, 'JKL', NULL),

(104, 'GHJ', 'P567'),

(105, 'UJM', 'P189'),

(106, 'UJM', NULL),

(107, 'RGJ', 'P567'),

(108, 'IJH', NULL),

(109, 'NMH', 'P547'),

(110, 'SDF', 'P189'),

(111, 'TGB', 'P546'),

(112, 'HUF', NULL),

(113, 'SDF', 'P789'),

(114, 'RGJ', 'P567'),

(115, 'JKG', 'P189');

**--Queries**

**-- 1. Find all the projects which are not currently taken up by any employee**

SELECT Pname

FROM Project

WHERE Pid NOT IN (SELECT Pid

FROM Employee

WHERE Pid IS NOT NULL);

**-- 2. Find all the other employees who are working in the same project with ‘ABC’**

SELECT Ename

FROM Employee

WHERE Pid IN (SELECT Pid

FROM Employee

WHERE Ename='ABC');

**-- 3. Find all the project names and No of employees working on each project**

SELECT Pname, COUNT(Employee.Pid)

FROM Project

LEFT JOIN Employee

ON Employee.Pid=Project.Pid

GROUP BY Project.Pid;

**-- 4. Find all the projects which have 2 or more employees with same name**

SELECT Pname

FROM Project

JOIN Employee

ON Project.Pid = Employee.Pid

GROUP BY Employee.Ename, Project.Pid

HAVING COUNT(Employee.Ename AND Employee.Pid) > 1;

**-- 5. List out the names of both employees and projects in alphabetical order into the single result.**

SELECT Pname, Ename

FROM Project

JOIN Employee

ON Project.Pid=Employee.Pid

ORDER BY Pname, Ename;

**Question 2:**

**Table Name : Products** (ProductID is the Primary Key, CategoryID is the foreign Key)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ProductID** | **ProductName** | **SupplierID** | **CategoryID** | **Unit** | **Price** |
| 1 | Chais | 1 | 1 | 200 | 18 |
| 2 | Chang | 1 | 1 | 120 | 19 |
| 3 | Aniseed Syrup | 1 | 2 | 50 | 10 |
| 4 | Chef Anton's Cajun Seasoning | 2 | 2 | 45 | 22 |
| 5 | Chef Anton's Gumbo Mix | 2 | 4 | 65 | 25 |
| 6 | Grandma's Boysenberry Spread | 3 | 2 | 78 | 25 |
| 7 | Uncle Bob's Organic Dried Pears | 3 | 3 | 98 | 30 |
| 8 | Northwoods Cranberry Sauce | 3 | 2 | 150 | 40 |
| 9 | Mishi Kobe Niku | 4 | 4 | 230 | 97 |
| 10 | Ikura | 4 | 5 | 45 | 31 |

**Table Name : Categories** (CategoryID is the Primary Key)

|  |  |  |
| --- | --- | --- |
| **CategoryID** | **CategoryName** | **Description** |
| 1 | Beverages | Soft drinks, coffees, teas, beers, and ales |
| 2 | Condiments | Sweet and savory sauces, relishes, spreads, and seasonings |
| 3 | Confections | Desserts, candies, and sweet breads |
| 4 | Dairy Products | Cheeses |
| 5 | Grains/Cereals | Breads, crackers, pasta, and cereal |

**--Creating Tables**

CREATE TABLE Categories (

CategoryID INT,

CategoryName VARCHAR(15),

Description VARCHAR(70),

PRIMARY KEY (CategoryID)

);

CREATE TABLE Products (

ProductID INT,

ProductName VARCHAR(40),

SupplierID INT,

CategoryID INT,

Unit INT,

Price INT,

PRIMARY KEY (ProductID),

FOREIGN KEY (CategoryID) REFERENCES Categories(CategoryID)

);

**--Inserting Values in Tables**

INSERT INTO Categories (CategoryID, CategoryName, Description)

VALUES (1, 'Beverages', 'Soft drinks, coffees, teas, beers, and ales'),

(2, 'Condiments', 'Sweet and savory sauces, relishes, spreads, and seasonings'),

(3, 'Confections', 'Desserts, candies, and sweet breads'),

(4, 'Dairy Products', 'Cheeses'),

(5, 'Grains/Cereals', 'Breads, crackers, pasta, and cereal');

INSERT INTO Products (ProductID, ProductName, SupplierID, CategoryID, Unit, Price)

VALUES (1, 'Chais', 1, 1, 200, 18),

(2, 'Chang', 1, 1, 120, 19),

(3, 'Aniseed Syrup', 1, 2, 50, 10),

(4, 'Chef Anton''s Cajun Seasoning', 2, 2, 45, 22),

(5, 'Chef Anton''s Gumbo Mix ', 2, 4, 65, 25),

(6, 'Grandma''s Boysenberry Spread', 3, 2, 78, 25),

(7, 'Uncle Bob''s Organic Dried Pears', 3, 3, 98, 30),

(8, 'Northwoods Cranberry Sauce', 3, 2, 150, 40),

(9, 'Mishi Kobe Niku', 4, 4, 230, 97),

(10, 'Ikura', 4, 5, 45, 31);

**--Queries**

**--1. Find all records where product name contains ‘ch’**

SELECT \*

FROM Products

WHERE ProductName LIKE '%ch%'

**--2. Find all records where price is greater than the average price of all products**

SELECT \*

FROM Products

WHERE Price>(SELECT AVG(Price) FROM Products);

**--3. Find all products whose category name is ‘Condiments’**

SELECT \*

FROM Products

WHERE CategoryID = (SELECT CategoryID

FROM Categories

WHERE CategoryName='Condiments');

**--4. Find No of products of each category**

SELECT CategoryName, COUNT(ProductID)

FROM Products

JOIN Categories

ON Products.CategoryID=Categories.CategoryID

GROUP BY CategoryName;

OR

SELECT CategoryID, COUNT(ProductID)

FROM Products

GROUP BY CategoryID;

**-- 5. The following query is too slow on the products table containing more than 10 lacs records**

**-- Select \* from products where Unit > 500;**

**-- Write a query that decreases the retrieval time from now**

**-- Hint: Create INDEX unit\_index on Products (Unit)**

CREATE INDEX unit\_index

ON Products (Unit);

SELECT \*

FROM Products

WHERE Unit > 500;