```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        PROJECT - HR DATABASE MAMAGEMENT SYSTEM
        BY 'SATYENDRA HOMKAR'
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
CREATE DATABASE HR_DATABASE_MANAGEMENT_SYSTEM_BY_SATYENDRA_HOMKAR
USE HR_DATABASE_MANAGEMENT_SYSTEM_BY_SATYENDRA_HOMKAR
--------------------------------------------------------------------------------------
CREATE TABLE regions (
region_id INT PRIMARY KEY,
region_name VARCHAR (25) DEFAULT NULL);
--------------------------------------------------------------------------------------
CREATE TABLE countries (
country_id CHAR (2) PRIMARY KEY,
country_name VARCHAR (40) DEFAULT NULL,
region_id INT NOT NULL,
FOREIGN KEY (region_id) REFERENCES regions (region_id) ON DELETE
CASCADE ON UPDATE CASCADE);
--------------------------------------------------------------------------------------
CREATE TABLE locations (
location_id INT PRIMARY KEY,
street_address VARCHAR (40) DEFAULT NULL,
postal_code VARCHAR (12) DEFAULT NULL,
city VARCHAR (30) NOT NULL,
state_province VARCHAR (25) DEFAULT NULL,
country_id CHAR (2) NOT NULL,
FOREIGN KEY (country_id) REFERENCES countries (country_id) ON DELETE
CASCADE ON UPDATE CASCADE);
--------------------------------------------------------------------------------------
CREATE TABLE jobs (
job_id INT PRIMARY KEY,
job_title VARCHAR (35) NOT NULL,
min_salary DECIMAL (8, 2) DEFAULT NULL,
max_salary DECIMAL (8, 2) DEFAULT NULL);
--------------------------------------------------------------------------------------
CREATE TABLE departments (
department_id INT PRIMARY KEY,
department_name VARCHAR (30) NOT NULL,
location_id INT DEFAULT NULL,
FOREIGN KEY (location_id) REFERENCES locations (location_id) ON DELETE
CASCADE ON UPDATE CASCADE);
--------------------------------------------------------------------------------------
CREATE TABLE employees (
employee_id INT PRIMARY KEY,
first_name VARCHAR (20) DEFAULT NULL,
last_name VARCHAR (25) NOT NULL,
email VARCHAR (100) NOT NULL,
phone_number VARCHAR (20) DEFAULT NULL,
hire_date DATE NOT NULL,
job_id INT NOT NULL,
```

```
salary DECIMAL (8, 2) NOT NULL,
manager_id INT DEFAULT NULL,
department_id INT DEFAULT NULL,
FOREIGN KEY (job_id) REFERENCES jobs (job_id) ON DELETE CASCADE ON
UPDATE CASCADE,
FOREIGN KEY (department_id) REFERENCES departments (department_id) ON
DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (manager_id) REFERENCES employees (employee_id))
-------------------------------------------------------------------------------
CREATE TABLE dependents (
dependent_id INT PRIMARY KEY,
first_name VARCHAR (50) NOT NULL,
last_name VARCHAR (50) NOT NULL,
relationship VARCHAR (25) NOT NULL,
employee_id INT NOT NULL,
FOREIGN KEY (employee_id) REFERENCES employees (employee_id) ON DELETE
CASCADE ON UPDATE CASCADE);
-------------------------------------------------------------------------------
INSERT INTO regions(region_id,region_name)
VALUES (1,'Europe'),
(2,'Americas'),
(3,'Asia'),
(4,'Middle East and Africa');
-------------------------------------------------------------------------------
INSERT INTO countries(country_id,country_name,region_id)
VALUES ('AR','Argentina',2), ('AU','Australia',3),
('BE','Belgium',1), ('BR','Brazil',2),
('CA','Canada',2), ('CH','Switzerland',1),
('CN','China',3),  ('DE','Germany',1),
('DK','Denmark',1), ('EG','Egypt',4),
('FR','France',1), ('HK','HongKong',3),
('IL','Israel',4), ('IN','India',3),
('IT','Italy',1), ('JP','Japan',3),
('KW','Kuwait',4), ('MX','Mexico',2),
('NG','Nigeria',4), ('NL','Netherlands',1),
('SG','Singapore',3), ('UK','United Kingdom',1),
('US','United States of America',2), ('ZM','Zambia',4),
('ZW','Zimbabwe',4);
-------------------------------------------------------------------------------
INSERT INTO locations(location_id,street_address,postal_code,city,state_province,
country_id)
VALUES (1400,'2014 Jabberwocky Rd','26192','Southlake','Texas','US'),
(1500,'2011 Interiors Blvd','99236','South San Francisco','California','US'),
(1700,'2004 Charade Rd','98199','Seattle','Washington','US'),
(1800,'147 Spadina Ave','M5V 2L7','Toronto','Ontario','CA'),
(2400,'8204 Arthur St',NULL,'London',NULL,'UK'),
(2500,'Magdalen Centre, The Oxford Science Park','OX9
  9ZB','Oxford','Oxford','UK'),
(2700,'Schwanthalerstr. 7031','80925','Munich','Bavaria','DE');
```

```
--------------------------------------------------------------------------------
INSERT INTO jobs(job_id,job_title,min_salary,max_salary)
VALUES (1,'Public Accountant',4200.00,9000.00),
(2,'Accounting Manager',8200.00,16000.00),
(3,'Administration Assistant',3000.00,6000.00),
(4,'President',20000.00,40000.00),
(5,'Administration Vice President',15000.00,30000.00),
(6,'Accountant',4200.00,9000.00),
(7,'Finance Manager',8200.00,16000.00),
(8,'Human Resources Representative',4000.00,9000.00),
(9,'Programmer',4000.00,10000.00),
(10,'Marketing Manager',9000.00,15000.00),
(11,'Marketing Representative',4000.00,9000.00),
(12,'Public Relations Representative',4500.00,10500.00),
(13,'Purchasing Clerk',2500.00,5500.00),
(14,'Purchasing Manager',8000.00,15000.00),
(15,'Sales Manager',10000.00,20000.00),
(16,'Sales Representative',6000.00,12000.00),
(17,'Shipping Clerk',2500.00,5500.00),
(18,'Stock Clerk',2000.00,5000.00),
(19,'Stock Manager',5500.00,8500.00);
--------------------------------------------------------------------------------
INSERT INTO departments(department_id,department_name,location_id)
VALUES (1,'Administration',1700),
(2,'Marketing',1800),
(3,'Purchasing',1700),
(4,'Human Resources',2400),
(5,'Shipping',1500),
(6,'IT',1400),
(7,'Public Relations',2700),
(8,'Sales',2500),
(9,'Executive',1700),
(10,'Finance',1700),
(11,'Accounting',1700);
--------------------------------------------------------------------------------
INSERT INTO employees
  (employee_id,first_name,last_name,email,phone_number,hire_date,
job_id,salary,manager_id,department_id)
VALUES (104,'Bruce','Ernst','bruce.ernst@sqltutorial.org','590.423.4568',
'1991-05-21',9,6000.00,103,6),
(100,'Steven','King','steven.king@sqltutorial.org','515.123.4567',
'1987-06-17',4,24000.00,NULL,9),
(101,'Neena','Kochhar','neena.kochhar@sqltutorial.org','515.123.4568',
'1989-09-21',5,17000.00,100,9),
(102,'Lex','De Haan','lex.de haan@sqltutorial.org','515.123.4569',
'1993-01-13',5,17000.00,100,9),
(103,'Alexander','Hunold','alexander.hunold@sqltutorial.org','590.423.4567',
'1990-01-03',9,9000.00,102,6),
(105,'David','Austin','david.austin@sqltutorial.org','590.423.4569',
```

```
'1997-06-25',9,4800.00,103,6),
(106,'Valli','Pataballa','valli.pataballa@sqltutorial.org','590.423.4560',
'1998-02-05',9,4800.00,103,6),
(107,'Diana','Lorentz','diana.lorentz@sqltutorial.org','590.423.5567',
'1999-02-07',9,4200.00,103,6),
(108,'Nancy','Greenberg','nancy.greenberg@sqltutorial.org','515.124.4569',
'1994-08-17',7,12000.00,101,10),
(109,'Daniel','Faviet','daniel.faviet@sqltutorial.org','515.124.4169',
'1994-08-16',6,9000.00,108,10),
(116,'Shelli','Baida','shelli.baida@sqltutorial.org','515.127.4563',
'1997-12-24',13,2900.00,114,3),
(117,'Sigal','Tobias','sigal.tobias@sqltutorial.org','515.127.4564',
'1997-07-24',13,2800.00,114,3),
(118,'Guy','Himuro','guy.himuro@sqltutorial.org','515.127.4565',
'1998-11-15',13,2600.00,114,3),
(119,'Karen','Colmenares','karen.colmenares@sqltutorial.org','515.127.4566',
'1999-08-10',13,2500.00,114,3),
(120,'Matthew','Weiss','matthew.weiss@sqltutorial.org','650.123.1234',
'1996-07-18',19,8000.00,100,5),
(121,'Adam','Fripp','adam.fripp@sqltutorial.org','650.123.2234',
'1997-04-10',19,8200.00,100,5),
(122,'Payam','Kaufling','payam.kaufling@sqltutorial.org','650.123.3234',
'1995-05-01',19,7900.00,100,5),
(123,'Shanta','Vollman','shanta.vollman@sqltutorial.org','650.123.4234',
'1997-10-10',19,6500.00,100,5),
(126,'Irene','Mikkilineni','irene.mikkilineni@sqltutorial.org','650.124.1224',
'1998-09-28',18,2700.00,120,5),
(145,'John','Russell','john.russell@sqltutorial.org',NULL,
'1996-10-01',15,14000.00,100,8),
(146,'Karen','Partners','karen.partners@sqltutorial.org',NULL,
'1997-01-05',15,13500.00,100,8),
(176,'Jonathon','Taylor','jonathon.taylor@sqltutorial.org',NULL,
'1998-03-24',16,8600.00,100,8),
(177,'Jack','Livingston','jack.livingston@sqltutorial.org',NULL,
'1998-04-23',16,8400.00,100,8),
(178,'Kimberely','Grant','kimberely.grant@sqltutorial.org',NULL,
'1999-05-24',16,7000.00,100,8),
(179,'Charles','Johnson','charles.johnson@sqltutorial.org',NULL,
'2000-01-04',16,6200.00,100,8),
(192,'Sarah','Bell','sarah.bell@sqltutorial.org','650.501.1876',
'1996-02-04',17,4000.00,123,5),
(193,'Britney','Everett','britney.everett@sqltutorial.org','650.501.2876',
'1997-03-03',17,3900.00,123,5),
(200,'Jennifer','Whalen','jennifer.whalen@sqltutorial.org','515.123.4444',
'1987-09-17',3,4400.00,101,1),
(201,'Michael','Hartstein','michael.hartstein@sqltutorial.org','515.123.5555',
'1996-02-17',10,13000.00,100,2),
(202,'Pat','Fay','pat.fay@sqltutorial.org','603.123.6666',
'1997-08-17',11,6000.00,201,2),
```

```
(203,'Susan','Mavris','susan.mavris@sqltutorial.org','515.123.7777',
'1994-06-07',8,6500.00,101,4),
(204,'Hermann','Baer','hermann.baer@sqltutorial.org','515.123.8888',
'1994-06-07',12,10000.00,101,7),
(205,'Shelley','Higgins','shelley.higgins@sqltutorial.org','515.123.8080',
'1994-06-07',2,12000.00,101,11);
-------------------------------------------------------------------------------
INSERT INTO dependents(dependent_id,first_name,last_name,relationship,employee_id)
VALUES (1,'Penelope','Gietz','Child',206),
(2,'Nick','Higgins','Child',205),
(3,'Ed','Whalen','Child',200),
(4,'Jennifer','King','Child',100),
(5,'Johnny','Kochhar','Child',101),
(6,'Bette','De Haan','Child',102),
(7,'Grace','Faviet','Child',109),
(8,'Matthew','Chen','Child',110),
(9,'Joe','Sciarra','Child',111),
(10,'Christian','Urman','Child',112),
(11,'Zero','Popp','Child',113),
(12,'Karl','Greenberg','Child',108),
(13,'Uma','Mavris','Child',203),
(14,'Vivien','Hunold','Child',103),
(15,'Cuba','Ernst','Child',104),
(16,'Fred','Austin','Child',105),
(17,'Helen','Pataballa','Child',106),
(18,'Dan','Lorentz','Child',107),
(19,'Bob','Hartstein','Child',201),
(20,'Lucille','Fay','Child',202),
(21,'Kirsten','Baer','Child',204),
(22,'Elvis','Khoo','Child',115),
(23,'Sandra','Baida','Child',116),
(24,'Cameron','Tobias','Child',117),
(25,'Kevin','Himuro','Child',118),
(26,'Rip','Colmenares','Child',119),
(27,'Julia','Raphaely','Child',114),
(28,'Woody','Russell','Child',145),
(29,'Alec','Partners','Child',146),
(30,'Sandra','Taylor','Child',176);
-------------------------------------------------------------------------------
/*TASK 1- 1)WRITE A QUERY FOR SELECT STATEMENTS :-
A. To get data from all the rows and columns in the employees table:*/
--ANS-
SELECT * FROM employees
-------------------------------------------------------------------------------
/*B.select data from the employee id, first name, last name, and hire date of
all rows in the employees table:*/
--Ans-
SELECT employee_id,first_name, last_name, hire_date FROM employees
-------------------------------------------------------------------------------
```

```
/*C. to get the first name, last name, salary, and new salary:
D. Increase the salary two times and named as New_SALARY from employees table */
--Ans-
SELECT first_name, last_name, salary,salary*2 as 'New Salary'
FROM employees
-------------------------------------------------------------------------------
/* 2) WRITE A QUERY FOR ORDER BY STATEMENTS :-
A. returns the data from the employee id, first name, last name, hire date,
and salary column of the employees table:*/
-- Ans
SELECT first_name, last_name, hire_date, salary
FROM employees
-------------------------------------------------------------------------------
/*B. to sort employees by first names in alphabetical order:*/
--Ans
SELECT  first_name FROM employees
ORDER BY first_name
-------------------------------------------------------------------------------
/*C. to sort the employees by the first name in ascending order and the last
name in descending order:*/
--Ans-
SELECT  first_name, last_name FROM employees
ORDER BY first_name ASC, last_name DESC
-------------------------------------------------------------------------------
/*D. to sort employees by salary from high to low:*/
--Ans-
SELECT first_name, last_name, salary
FROM employees
ORDER BY salary DESC
-------------------------------------------------------------------------------
/*E. to sort the employees by values in the hire_date column from:*/
--Ans-
SELECT first_name, last_name, hire_date
FROM employees
ORDER BY hire_date
-------------------------------------------------------------------------------
/*F. sort the employees by the hire dates in descending order: */
--Ans-
SELECT first_name, last_name, hire_date
FROM employees
ORDER BY hire_date DESC
-------------------------------------------------------------------------------
/* 3)WRITE A QUERY FOR DISTINCT STATEMENTS :-
A. selects the salary data from the salary column of the employees table and
sorts them from high to low:*/
--Ans-
SELECT salary FROM employees
ORDER BY salary DESC
-------------------------------------------------------------------------------
```

```
/*B. select unique values from the salary column of the employees table:*/
--Ans-
SELECT  DISTINCT salary FROM employees
-------------------------------------------------------------------------------
/*C. selects the job id and salary from the employees table:*/
--Ans-
SELECT job_id, salary FROM employees
-------------------------------------------------------------------------------
/*D. to remove the duplicate values in job id and salary:*/
--Ans-
SELECT DISTINCT job_id,
salary FROM employees
-------------------------------------------------------------------------------
/*E. returns the distinct phone numbers of employees:*/
--Ans-
SELECT DISTINCT phone_number
FROM employees
-------------------------------------------------------------------------------
/* 4)WRITE A QUERY FOR TOP N STATEMENTS :-
A. returns all rows in the employees table sorted by the first_name column.*/
--Ans-
SELECT * FROM employees
ORDER BY first_name
-------------------------------------------------------------------------------
/*B. to return the first 5 rows in the result set returned by the SELECT clause:*/
--Ans-
SELECT TOP (5)* FROM employees
ORDER BY first_name
-------------------------------------------------------------------------------
/*C. to return five rows starting from the 4th row:*/
--Ans-
SELECT * FROM employees
ORDER BY first_name
OFFSET 3 ROWS
FETCH NEXT 5 ROWS ONLY
-------------------------------------------------------------------------------
/*D. gets the top five employees with the highest salaries.*/
--Ans-
SELECT TOP (5)* FROM employees
ORDER BY salary DESC
-------------------------------------------------------------------------------
/*E. to get employees who have the 2nd highest salary in the company*/
--Ans-
SELECT * FROM employees
WHERE salary= (SELECT MAX(salary) FROM employees
            WHERE salary<(SELECT MAX(salary) FROM employees))
-------------------------------------------------------------------------------
/* 5)WRITE A QUERY FOR WHERE CLAUSE and COMPARISON OPERATORS :-
A.finds employees who have salaries greater than 14,000 and sorts the results sets
```

```sql
based on the salary in descending order.*/
--Ans-
SELECT * FROM employees
    WHERE salary>14000
    ORDER BY salary DESC
-------------------------------------------------------------------------------
/*B.finds all employees who work in the department id 5.*/
--Ans-
SELECT * FROM employees
    WHERE department_id=5
-------------------------------------------------------------------------------
/*C.finds the employee whose last name is Chen*/
--Ans-
SELECT * FROM employees
    WHERE last_name='Chen'
-------------------------------------------------------------------------------
/*D.get all employees who joined the company after January 1st, 1999*/
--Ans-
SELECT * FROM employees
    WHERE hire_date> '1999-01-01'
-------------------------------------------------------------------------------
/*E. find the employees who joined the company in 1999,*/
--Ans-
SELECT * FROM employees
    WHERE DATENAME(YEAR,hire_date)= 1999
-------------------------------------------------------------------------------
/*F.finds the employee whose last name is Himuro*/
--Ans-
SELECT * FROM employees
    WHERE last_name='Himuro'
-------------------------------------------------------------------------------
/*H.find all employees who do not have phone numbers:*/
--Ans-
SELECT * FROM employees
    WHERE phone_number is null
-------------------------------------------------------------------------------
/*I. returns all employees whose department id is not 8.*/
--Ans-
SELECT * FROM employees
    WHERE department_id !=8
-------------------------------------------------------------------------------
/*J. finds all employees whose department id is not eight and ten.*/
--Ans-
SELECT * FROM employees
    WHERE department_id!=8 and department_id!=10
-------------------------------------------------------------------------------
/*K.find the employees whose salary is greater than 10,000,*/
--Ans-
SELECT * FROM employees
```

```
    WHERE salary>10000
-------------------------------------------------------------------------------
/*L. finds employees in department 8 and have the salary greater than 10,000:*/
--Ans-
SELECT * FROM employees
    WHERE department_id=8 AND salary>10000
-------------------------------------------------------------------------------
/*M. returns all employees whose salaries are less than 10,000:*/
--Ans-
SELECT * FROM employees
    WHERE salary<10000
-------------------------------------------------------------------------------
/*N. finds employees whose salaries are greater than or equal 9,000:*/
--Ans-
SELECT * FROM employees
    WHERE salary>=9000
-------------------------------------------------------------------------------
/*O. finds employees whose salaries are less than or equal to 9,000:*/
--Ans-
SELECT * FROM employees
    WHERE salary<=9000
-------------------------------------------------------------------------------
/*6)WRITE A QUERY FOR:-Create table Course with column course_id and cousre_name
  */
--Ans-
CREATE TABLE course (
course_id INT,
course_name VARCHAR(100)
);
-------------------------------------------------------------------------------
/*A. adds a new column named credit_hours to the courses table.*/
--Ans-
ALTER TABLE course
ADD credit_hours INT
-------------------------------------------------------------------------------
/*B. adds the fee and max_limit columns to the courses table and places these
columns after the course_name column.*/
--Ans-
ALTER TABLE course
ADD fee FLOAT, max_limit FLOAT
-------------------------------------------------------------------------------
/*C. changes the attribute of the fee column to NOT NULL.*/
--Ans-
ALTER TABLE course
ALTER COLUMN fee FLOAT NOT NULL
-------------------------------------------------------------------------------
/*D. to remove the fee column of the courses table*/
--Ans-
ALTER TABLE course
```

```
DROP COLUMN Fee
-------------------------------------------------------------------------------
/*E. removes the max_limit and credit_hours of the courses table.*/
--Ans-
ALTER TABLE course
DROP COLUMN max_limit
-------------------------------------------------------------------------------
--SQL FOREIGN KEY constraint
CREATE TABLE projects (
 project_id INT PRIMARY KEY,
 project_name VARCHAR(255),
 start_date DATE NOT NULL,
 end_date DATE NOT NULL
);
-------------------------------------------------------------------------------
CREATE TABLE project_milestones(
 milestone_id INT PRIMARY KEY,
 project_id INT,
 milestone_name VARCHAR(100)
);
-------------------------------------------------------------------------------
/*A. to add an SQL FOREIGN KEY constraint to the project_milestones table to
  enforce
the relationship between the projects and project_milestones tables.*/
--Ans-
ALTER TABLE project_milestones
ADD CONSTRAINT FK_Project
FOREIGN KEY (project_id) REFERENCES  projects(project_id);
-------------------------------------------------------------------------------
/*TASK 2:
Logical Operators and Special Operators
A. finds all employees whose salaries are greater than 5,000 and less than
  7,000:*/
--Ans-
SELECT * FROM employees
    WHERE salary>5000 AND salary<7000
-------------------------------------------------------------------------------
/*B. finds employees whose salary is either 7,000 or 8,000:*/
--Ans-
SELECT * FROM employees
    WHERE salary=7000 OR salary=8000
-------------------------------------------------------------------------------
/*C. finds all employees who do not have a phone number:*/
--Ans-
SELECT * FROM employees
    WHERE phone_number IS NULL
-------------------------------------------------------------------------------
/*D. finds all employees whose salaries are between 9,000 and 12,000.*/
--Ans
```

```sql
SELECT * FROM employees
    WHERE salary  between 9000 AND 12000
-------------------------------------------------------------------------------
/*E. finds all employees who work in the department id 8 or 9.*/
--Ans-
SELECT * FROM employees
    WHERE department_id= 8 OR department_id=9
--OR
SELECT * FROM employees
    WHERE department_id BETWEEN 8 AND 9
-------------------------------------------------------------------------------
/*F. finds all employees whose first name starts with the string jo*/
--Ans-
 SELECT * FROM employees
    WHERE first_name LIKE 'JO%'
-------------------------------------------------------------------------------
/*G. finds all employees with the first names whose the second character is h*/
--Ans-
SELECT * FROM employees
    WHERE first_name LIKE '_h%'
-------------------------------------------------------------------------------
/*H. finds all employees whose salaries are greater than all salaries of employees

in the department 8:*/
--Ans-
SELECT *FROM employees
    WHERE salary>( SELECT MAX(salary) FROM employees
                   WHERE department_id=8)
-------------------------------------------------------------------------------
/*Part 2:-
A. finds all employees whose salaries are greater than the average salary of
every department:*/
--Ans-
SELECT * FROM employees
    WHERE Salary> ( SELECT AVG(salary) FROM employees)
-------------------------------------------------------------------------------
/*B. finds all employees who have dependents:*/
--Ans-
SELECT * FROM employees
    WHERE employee_id IN (SELECT employee_id FROM dependents
                          WHERE dependents.employee_id= employees.employee_id)
-- OR----
SELECT *FROM employees e
JOIN dependents d ON e.employee_id=d.employee_id
-------------------------------------------------------------------------------
/*C.find all employees whose salaries are between 2,500 and 2,900:*/
--Ans-
SELECT * FROM employees
    WHERE salary BETWEEN 2500 AND 2900
```

```
-------------------------------------------------------------------------------
/*D.to find all employees whose salaries are not in the range of 2,500 & 2,900:*/
--Ans-
SELECT * FROM employees
    WHERE salary  NOT BETWEEN 2500 AND 2900
-------------------------------------------------------------------------------
/*E. to find all employees who joined the company between January 1,1999, &
December 31,2000:*/
--Ans-
SELECT * FROM employees
    WHERE hire_date BETWEEN '1999-01-01' AND '2000-12-31'
-------------------------------------------------------------------------------
/*F. to find employees who have not joined the company from January 1, 1989 to
December 31, 1999:*/
--Ans-
SELECT * FROM employees
    WHERE hire_date  NOT BETWEEN '1989-01-01' AND '1999-12-31'
-------------------------------------------------------------------------------
/*G. to find employees who joined the company between 1990 and 1993:*/
--Ans-
SELECT * FROM employees
    WHERE DATENAME(YEAR,hire_date) BETWEEN 1990 AND 1993
-------------------------------------------------------------------------------
/*Part 3:-
A.find all employees whose first names start with Da*/
--Ans-
SELECT * FROM employees
    WHERE first_name LIKE 'Da%'
-------------------------------------------------------------------------------
/*B.find all employees whose first names end with er*/
--Ans-
SELECT * FROM employees
    WHERE first_name LIKE '%er'
-------------------------------------------------------------------------------
/*C.find employees whose last names contain the word an:*/
--Ans-
SELECT * FROM employees
    WHERE last_name LIKE '%an%'
-------------------------------------------------------------------------------
/*D. retrieves employees whose first names start with Jo and are followed by at
most 2 characters:*/
--Ans-
SELECT * FROM employees
    WHERE first_name LIKE 'Jo__'
-------------------------------------------------------------------------------
/*E.find employees whose first names start with any number of characters and are
followed by at most one character:*/
--Ans-
SELECT * FROM employees
```

```
    WHERE first_name LIKE '%_'
-------------------------------------------------------------------------------
/*F.to find all employees whose first names start with the letter S but not start
 with Sh:*/
--Ans-
SELECT * FROM employees
    WHERE first_name  NOT LIKE  'SH%' AND first_name LIKE 'S%'
-------------------------------------------------------------------------------
/*Part 4:-
A. retrieves all employees who work in the department id 5.*/
--Ans-
SELECT * FROM employees
    WHERE department_id=5
-------------------------------------------------------------------------------
/*B. To get the employees who work in the department id 5 and with a salary not
greater than 5000.*/
--Ans
SELECT * FROM employees
    WHERE department_id=5 AND salary <=5000
-------------------------------------------------------------------------------
/*C. statement gets all the employees who are not working in the departments
1, 2, or 3.*/
--Ans-
SELECT * FROM employees
    WHERE department_id NOT IN(1,2,3)
-------------------------------------------------------------------------------
/*D.retrieves all the employees whose first names do not start with the letter
  D.*/
--Ans-
SELECT * FROM employees
    WHERE first_name NOT LIKE 'D%'
-------------------------------------------------------------------------------
/*E. to get employees whose salaries are not between 5,000 and 1,000*/
--Ans-
SELECT * FROM employees
    WHERE salary NOT BETWEEN 1000 AND 5000
-------------------------------------------------------------------------------
/*  TASK 3:JOINS:-
1) Write a Query to
A. To get the information of the department id 1,2, and 3*/
--Ans-
SELECT * FROM departments
    WHERE department_id IN (1,2,3)
-------------------------------------------------------------------------------
/*B.To get the information of employees who work in the department id 1, 2 and 3*/
--Ans-
SELECT * FROM employees e
JOIN departments d ON e.department_id=d.department_id
    WHERE e.department_id IN (1,2,3)
```

```sql
-------------------------------------------------------------------------------
/*A. To query the country names of US, UK, and China*/
--Ans-
SELECT * FROM countries
    WHERE country_name IN('United States of America','United Kingdom','China')
-------------------------------------------------------------------------------
/*B. query retrieves the locations located in the US, UK and China:*/
--Ans-
SELECT * FROM countries c
JOIN locations i On c.country_id= i.country_id
    WHERE country_name IN('United States of America','United Kingdom','China')
-------------------------------------------------------------------------------
/*C. To join the countries table with the locations table*/
--Ans-
SELECT * FROM countries c
JOIN locations i On c.country_id= i.country_id
-------------------------------------------------------------------------------
/*D.To find the country that does not have any locations in the locations table*/
--Ans-
SELECT * FROM countries c
LEFT JOIN locations i On c.country_id= i.country_id
    WHERE location_id IS NULL
-------------------------------------------------------------------------------
/*Write a query to join 3 tables: regions, countries, and locations*/
--Ans-
SELECT * FROM regions r
JOIN countries c ON r.region_id=c.region_id
JOIN locations l ON c.country_id=l.country_id
-------------------------------------------------------------------------------
/*The manager_id column specifies the manager of an employee. Write a query
statement to joins the employees table to itself to query the information of
 who reports to whom.*/
--Ans-
SELECT  m.first_name+' '+m.last_name As 'Manager Name',
        e.First_Name+ ' '+ e.Last_Name As'Emp Name'
FROM Employees e
JOIN Employees m ON e.Manager_ID = m.employee_id
-------------------------------------------------------------------------------
/* SQL FULL OUTER JOIN*/
CREATE TABLE fruits (
fruit_id INT PRIMARY KEY,
fruit_name VARCHAR (255) NOT NULL,
basket_id INTEGER);

CREATE TABLE baskets (
basket_id INT PRIMARY KEY,
basket_name VARCHAR (255) NOT NULL
);
INSERT INTO baskets (basket_id, basket_name)
```

```
VALUES
(1, 'A'),
(2, 'B'),
(3, 'C');
INSERT INTO fruits (
fruit_id, fruit_name,basket_id)
VALUES
(1, 'Apple', 1),
(2, 'Orange', 1),
(3, 'Banana', 2),
(4, 'Strawberry', NULL);
-------------------------------------------------------------------------------
/*A. Write a query to returns each fruit that is in a basket and each basket that
has a fruit, but also returns each fruit that is not in any basket and each basket
that does not have any fruit.*/
--Ans-
SELECT * FROM baskets b
FULL JOIN Fruits f ON b.basket_id =f.basket_id
-------------------------------------------------------------------------------
/*B. Write a query to find the empty basket, which does not store any fruit*/
SELECT * FROM baskets b
FULL JOIN Fruits f ON b.basket_id =f.basket_id
    WHERE fruit_id IS NULL
-------------------------------------------------------------------------------
/*C. Write a query which fruit is not in any basket*/
--Ans-
SELECT * FROM Fruits f
FULL JOIN baskets b ON b.basket_id =f.basket_id
    WHERE basket_name IS NULL
-------------------------------------------------------------------------------
--SQL GROUP BY clause-Write a Query
/*A. to group the values in department_id column of the employees table:*/
SELECT  Department_id,
sum(salary) as Salary_total FROM Employees
GROUP BY Department_id
-------------------------------------------------------------------------------
/*B. to count the number of employees by department:*/
--Ans-
SELECT department_id,
COUNT(Employee_id) As 'Total employee' FROM employees
GROUP BY department_id
-------------------------------------------------------------------------------
/*C. returns the number of employees by department*/
SELECT department_name,
COUNT(Employee_id) as 'Total Employee'
FROM Departments d
JOIN Employees e ON e.department_id=d.department_id
GROUP BY d.department_name
-------------------------------------------------------------------------------
```

```sql
/*D. to sort the departments by headcount:*/
SELECT department_name,
COUNT(Employee_id) as 'Headcount'
FROM Departments d
JOIN Employees e ON e.department_id=d.department_id
GROUP BY d.department_name
-------------------------------------------------------------------------------
/*E. to find departments with headcounts are greater than 5:*/
SELECT department_name,
COUNT(Employee_id) as 'Headcount'
FROM Departments d
JOIN Employees e ON e.department_id=d.department_id
GROUP BY d.department_name
HAVING COUNT(Employee_id) >5
-------------------------------------------------------------------------------
/*F.returns the minimum, maximum and avg salary of employees in each department*/
SELECT department_name,
MIN(salary) as 'minimum salary',
MAX(salary) as 'maximum salary',
AVG(salary) as 'Avg salary'
FROM Departments d
JOIN Employees e ON e.department_id=d.department_id
GROUP BY d.department_name
-------------------------------------------------------------------------------
/*G. To get the total salary per department.*/
SELECT department_name,
SUM(Salary) as 'Total Salary'
FROM Departments d
JOIN Employees e ON e.department_id=d.department_id
GROUP BY d.department_name
-------------------------------------------------------------------------------
/*. groups rows with the same values both department_id and job_id columns in the
same group then return the rows for each of these groups*/
SELECT d.department_id,
j.job_id,
COUNT(E. Employee_id) As 'Total Emp'
FROM departments d
JOIN Employees e ON d.department_id=e.department_id
JOIN Jobs j ON j.job_id=e.job_id
GROUP BY d.department_id, j.job_id
-------------------------------------------------------------------------------
/*SQL HAVING clause-Write a Query
A. To get the managers and their direct reports, and to group employees by the
managers and to count the direct reports.*/
--Ans-
SELECT  m.first_name+' '+m.last_name As 'Manager Name',
        COUNT(e.employee_id) AS' Total Emp Reporting'
FROM Employees e
JOIN Employees m ON e.Manager_ID = m.employee_id
```

```
GROUP BY m.first_name+' '+m.last_name
-------------------------------------------------------------------------------
/*B. To find the managers who have at least five direct reports*/
--Ans-
SELECT  m.first_name+' '+m.last_name As 'Manager Name',
        COUNT(e.employee_id) AS' Total Emp Reporting'
FROM Employees e
JOIN Employees m ON e.Manager_ID = m.employee_id
GROUP BY m.first_name+' '+m.last_name
HAVING COUNT(e.employee_id)>=5
-------------------------------------------------------------------------------
/*C. calculates the sum of salary that the company pays for each department and
selects only the departments with the sum of salary between 20000 and 30000.*/
SELECT department_name,
SUM(salary) as 'total salary'
FROM Departments d
JOIN Employees e ON d.department_id=e.department_id
GROUP BY department_name
HAVING SUM(salary)>=20000 and SUM(salary)<=30000
-------------------------------------------------------------------------------
/*D. To find the department that has employees with the lowest salary greater
than 10000*/
--Ans-
SELECT department_name,
min(salary) as 'Lowest salary'
FROM Departments d
JOIN Employees e ON d.department_id=e.department_id
GROUP BY department_name
HAVING min(salary)>10000
-------------------------------------------------------------------------------
/*E. To find the departments that have the average salaries of employees
between 5000 and 7000 */
SELECT department_name,
Avg(salary) as 'Avg salary'
FROM Departments d
JOIN Employees e ON d.department_id=e.department_id
GROUP BY department_name
HAVING Avg(salary) BETWEEN 5000 AND 7000
-------------------------------------------------------------------------------
/*TASK 5 (Other Queries)
Write a Query to combine the first name and last name of employees & dependents*/
SELECT first_name, Last_name FROM Employees
UNION
SELECT First_name, last_name FROM dependents
-------------------------------------------------------------------------------
/*SQL EXISTS operator
A. finds all employees who have at least one dependent.*/
--Ans-
SELECT e.Employee_ID, e.First_Name, e.Last_Name
```

```sql
FROM Employees e
WHERE EXISTS (
    SELECT 1
    FROM dependents d
    WHERE d.employee_id = e.Employee_id);
-------------------------------------------------------------------------------
/*B . finds employees who do not have any dependents:*/
SELECT e.Employee_ID, e.First_Name, e.Last_Name
FROM Employees e
WHERE  NOT EXISTS (
    SELECT 1
    FROM dependents d
    WHERE d.employee_id = e.Employee_id);
-------------------------------------------------------------------------------
/*SQL CASE expression
A. Suppose the current year is 2000. How to use the simple CASE expression to get
the work anniversaries of employees by*/
--Ans-
SELECT Employee_id, First_name, Last_name, YEAR(Hire_Date) AS Hire_Year,
    CASE YEAR(Hire_Date)
        WHEN 2000 THEN 'New Joining'
        WHEN 1999 THEN '1 Year Anniversary'
        WHEN 1998 THEN '2 Year Anniversary'
        WHEN 1997 THEN '3 Year Anniversary'
        WHEN 1996 THEN '4 Year Anniversary'
        WHEN 1995 THEN '5 Year Anniversary'
        ELSE 'cross 5th annivarsary'
    END AS Work_Anniversary
FROM Employees;
-------------------------------------------------------------------------------
/*B. Write a Query If the salary is less than 3000, the CASE expression returns
“Low”. If the salary is between 3000 and 5000, it returns “average”. When the
salary is greater than 5000, the CASE expression returns “High”.*/
SELECT Employee_id, First_name, Last_name, salary,
    CASE
        WHEN salary < 3000 THEN 'Low salary'
        WHEN  salary BETWEEN 3000 AND 5000 THEN 'Avg Salary'
        ELSE 'High Salary'
    END AS 'Salary Status'
FROM Employees;
-------------------------------------------------------------------------------
/*SQL UPDATE statement
Write a Query to update Sarah’s last name from Bell to Lopez
How to make sure that the last names of children are always matched with the
last name of parents in the employees table*/
--Ans-
UPDATE Employees
SET Last_Name = 'Lopez'
WHERE First_Name = 'Sarah' AND Last_Name = 'Bell'
```

```sql
CREATE TRIGGER trg_UpdateChildLastNames
ON Employees
AFTER UPDATE
AS
BEGIN
    UPDATE e
    SET e.Last_Name = d.Last_Name
    FROM Employees e
    INNER JOIN dependents d
        ON e.employee_id = d.employee_ID
    WHERE e.Last_Name <> e.Last_Name;
END;
-----------------------------------------------------------------------------
/*SQL SUBQUERY Write a Query :-
A. Combine Above two queries using subquery inorder find all departments located
at the location whose id is 1700 and find all employees that belong to the
  location
1700 by using the department id list of the previous query*/
SELECT *  FROM employees
WHERE department_id IN( SELECT Department_id FROM  departments
                        WHERE location_id = 1700)
-----------------------------------------------------------------------------
/*B. to find all employees who do not locate at the location 1700:*/
SELECT *  FROM employees
WHERE department_id IN( SELECT Department_id FROM  departments
                        WHERE location_id != 1700)

-----------------------------------------------------------------------------
/*C. finds the employees who have the highest salary:*/
SELECT first_name, last_name
FROM Employees
WHERE Salary = ( SELECT MAX(Salary) FROM employees)
-----------------------------------------------------------------------------
/*D. finds all employees who salaries are greater than the average salary of all
employees:*/
SELECT first_name, last_name, salary FROM employees
WHERE salary> ( SELECT AVG(Salary)FROM employees)
-----------------------------------------------------------------------------
/*E. finds all departments which have at least one employee with the salary is
greater than 10,000:*/
SELECT department_name from departments
WHERE department_id IN(SELECT department_id FROM employees
                        WHERE salary>10000)
-----------------------------------------------------------------------------
/*F. finds all departments that do not have any employee with the salary greater
than 10,000:*/ --9,8,2,11,10,3,7
SELECT department_name from departments
WHERE department_id NOT IN(SELECT DISTINCT Department_id FROM employees
```

```
                        WHERE salary>10000)
-------------------------------------------------------------------------------
/*G. to find the lowest salary by department:*/
SELECT department_name,
( SELECT MIN( Salary) FROM employees e
WHERE e.department_id=d.department_id
                GROUP BY department_id) As Salary
from departments d
-------------------------------------------------------------------------------
/*H. finds all employees whose salaries are greater than the lowest salary of
every department:*/
SELECT first_name, last_name FROM employees
WHERE Salary >( SELECT TOP(1)MIN (Salary)as minsalary FROM Employees
GROUP BY department_id order by minsalary DESC)
-------------------------------------------------------------------------------
/*I. finds all employees whose salaries are greater than or equal to the highest
salary of every department*/
SELECT first_name, last_name FROM employees
WHERE Salary >=( SELECT TOP(1)Max (Salary)as maxsalary FROM Employees
GROUP BY department_id order by maxsalary DESC)
-------------------------------------------------------------------------------
/*J. returns the average salary of every department*/
SELECT Department_name,
(SELECT AVG(Salary) FROM Employees e
WHERE e.department_id=d.department_id) As 'AVG salary'
FROM Departments d
-------------------------------------------------------------------------------
/*K. to calculate the average of average salary of departments :*/
SELECT AVG(avg_salary) AS avg_of_avg_salary
FROM (
    SELECT Department_ID, AVG(Salary) AS avg_salary
    FROM Employees
    GROUP BY Department_ID
) AS dept_avg;
-------------------------------------------------------------------------------
/*L. finds the salaries of all employees, their average salary, and the difference
between the salary of each employee and the average salary.*/
--Ans-
SELECT first_name, last_name, salary,
(SELECT AVG(salary) FROM Employees) As 'Avg Salary',
salary-(SELECT AVG(salary) FROM Employees) as ' diff AVG Salary'
FROM Employees
-------------------------------------------------------------------------------
-------------------------------------------------------------------------------
```