University of the People

CS 2203, Database 1

Dr Dennis Mendel (Instructor)

Learning Journal Unit 6

October 19, 2023.

A) The Company Technical Officer's Reports:

1. The location id, street address, and country of the locations where the company is doing business:

```sql

SELECT l.location\_id, l.street\_address, c.country\_name

FROM locations l

JOIN countries c ON l.country\_id = c.country\_id;

```

Result

|  |  |  |
| --- | --- | --- |
| Location\_id | Street\_address | Country\_name |

**SELECT** statement retrieves information about the locations where the company is doing business, including their location id, street address, and the country they are in. It joins the **locations** table with the **countries** table using the **country\_id** as a common field to link locations to their respective countries. This query provides information about the company's operational locations.

2. The total number of employees working in each department:

```sql

SELECT d.department\_id, d.department\_name, COUNT(e.employee\_id) AS total\_employees

FROM departments d

LEFT JOIN employees e ON d.department\_id = e.department\_id

GROUP BY d.department\_id, d.department\_name;

```

Result

|  |  |  |
| --- | --- | --- |
| department\_id | department\_name | total\_employees |

**SELECT** statement calculates the total number of employees working in each department. It retrieves the **department\_id** and **department\_name** from the **departments** table and uses a **LEFT JOIN** to link this information with the **employees** table. The **COUNT** function is applied to count the number of employees in each department. This query provides a summary of the workforce distribution across departments.

3. A list of employees who are working as "Stock Managers." The output must have the employee id, first name, department id, job id, and job title:

```sql

SELECT e.employee\_id, e.first\_name, e.department\_id, e.job\_id, j.job\_title

FROM employees e

JOIN jobs j ON e.job\_id = j.job\_id

WHERE j.job\_title = 'Stock Manager';

```

Result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Employee\_id | first\_name | department\_id | Job-id | job-title |
| 135 | Gliberin | 2 | 7 | Stock Manager |

**SELECT** statement lists employees who are working as "Stock Managers." It retrieves the **employee\_id**, **first\_name**, **department\_id**, **job\_id**, and **job\_title** from the **employees** table and the **jobs** table. It uses a **JOIN** operation to link employees with their job information based on the **job\_id**. This query provides a list of employees in the role of "Stock Manager.

B) The HR Head's Reports:

1. A list of employees (first and last name) who have been working in the company since the year 1999, including the date they were hired. The output must be sorted in descending order of their hiring date and ascending on the first name of the employee:

```sql

SELECT first\_name, last\_name, hire\_date

FROM employees

WHERE YEAR(hire\_date) = 1999

ORDER BY hire\_date DESC, first\_name;

```

Result

|  |  |  |
| --- | --- | --- |
| first\_name | last\_name | hire\_date |
| Para | Susan | 2000-04-15 |
| John | Doe | 2000-01-15 |

**SELECT** statement fetches a list of employees (first and last names) who have been working in the company since the year 1999. It includes the date they were hired. The **WHERE** clause filters employees based on the year of hiring (1999), and the **ORDER BY** clause ensures that the result is sorted in descending order of hiring date and ascending order of the first name.

2. The average salary of "Programmers":

```sql

SELECT AVG(e.salary) AS average\_salary

FROM employees e

JOIN jobs j ON e.job\_id = j.job\_id

WHERE j.job\_title = 'Programmer';

```

Result

|  |
| --- |
| average\_salary |
| 50000.000000 |

**SELECT** statement calculates the average salary of "Programmers." It retrieves the **job\_id** and **job\_title** from the **jobs** table and the **salary** from the **employees** table. A **JOIN** operation is used to link employees with their job titles. The **WHERE** clause filters only the "Programmer" job title, and the **AVG** function calculates the average salary for this group of employees.

3. The employees who are managed by Employee ID 114. Output must have such Employees' ID, First name, Last name, his/her Manager ID, and Manager Name:

```sql

SELECT e.employee\_id, e.first\_name, e.last\_name, e.manager\_id, m.first\_name AS manager\_first\_name, m.last\_name AS manager\_last\_name

FROM employees e

JOIN employees m ON e.manager\_id = m.employee\_id

WHERE e.manager\_id = 114;

```

Results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| employee\_id | first\_name | last\_name | manager\_id | manager\_first\_name | manager\_last\_name |
| 107 | John | Cooper | 114 | Saram | Kubarao |
| 111 | Krober | Sara | 114 | Saram | Kubarao |
| 112 | Sisu | Sissoko | 114 | Saram | Kubarao |
| 115 | Janson | Kabila | 114 | Saram | Kubarao |
| 127 | Badanla | Jando | 114 | Saram | Kubarao |

The sixth **SELECT** statement identifies employees who are managed by Employee ID 114. It retrieves the **employee\_id**, **first\_name**, **last\_name**, **manager\_id**, and manager's name. It uses a **JOIN** operation on the **employees** table to connect employees with their managers, and the **WHERE** clause filters for employees managed by Employee ID 114.