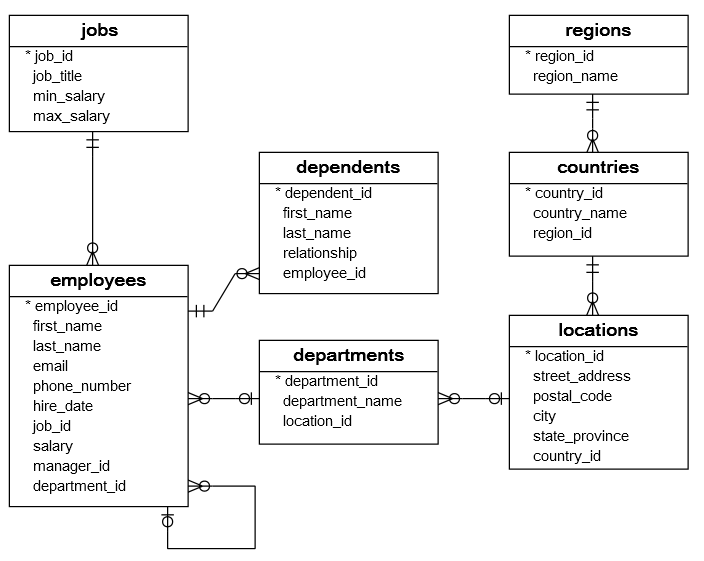
# **Learning Journal - Unit 6**

Computer Science, University of the People

CS 2203-01 Databases 1 - AY2024-T3

Instructor, Irfan Rashid Thoker

March 14, 2024

This week I was tasked with building scripts for different employees in a HR company. The sample database was taken from the website provided and the scripts were built per each requirement.

1. Technical Officer:
   * 1. List of Locations and Countries:

SELECT

location\_id,

street\_address,

countries.country\_name

FROM

locations INNER JOIN countries

ON locations.country\_id = countries.country\_id;

For this query what we do is join the table’s locations and countries using the country\_id column. I chose the INNER JOIN since I’m only interested in locations that have a corresponding country, any records that don’t have one will and should be ignored. But since the relationship doesn’t allow this there is no fear of this even happening.  
  
Result:

|  |  |  |
| --- | --- | --- |
| **location\_id** | **street\_address** | **country\_name** |
| 1400 | 2014 Jabberwocky Rd | United States of America |
| 1500 | 2011 Interiors Blvd | United States of America |
| 1700 | 2004 Charade Rd | United States of America |
| 1800 | 147 Spadina Ave | Canada |
| 2400 | 8204 Arthur St | United Kingdom |
| 2500 | Magdalen Centre, The Oxford Science Park | United Kingdom |
| 2700 | Schwanthalerstr. 7031 | Germany |

* + 1. Employee Count per Department

SELECT

department\_name,

COUNT(employee\_id) AS total\_employees

FROM

departments INNER JOIN employees

ON departments.department\_id = employees.department\_id

GROUP BY

department\_name;

For this report I grouped the values in the employees table by the ‘department\_name’ and then counted the number of employees in each group using the aggregate function ‘COUNT’. To be able to group by it and display it joined the ‘departments’ table with the ‘employees’ table so that I can display the name of the department instead of just the ‘department\_id’.

Result:

|  |  |
| --- | --- |
| **department\_name** | **total\_employees** |
| Accounting | 2 |
| Administration | 1 |
| Executive | 3 |
| Finance | 6 |
| Human Resources | 1 |
| IT | 5 |
| Marketing | 2 |
| Public Relations | 1 |
| Purchasing | 6 |
| Sales | 6 |
| Shipping | 7 |

* + 1. Identifying 'Stock Managers'

SELECT

employee\_id,

first\_name,

department\_id,

employees.job\_id,

job\_title

FROM

employees INNER JOIN jobs

ON employees.job\_id = jobs.job\_id

WHERE

employees.job\_id = 19;

For this report I joined the ‘employees’ table with the ‘jobs’ table using an INNER JOIN since every employee must have a ‘job\_id’ and that job\_id must exist in ‘jobs’ table here again there is no worry of there being missing records.

Result:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **employee\_id** | **first\_name** | **department\_id** | **job\_id** | **job\_title** |
| 120 | Matthew | 5 | 19 | Stock Manager |
| 121 | Adam | 5 | 19 | Stock Manager |
| 122 | Payam | 5 | 19 | Stock Manager |
| 123 | Shanta | 5 | 19 | Stock Manager |

1. HR Head:
   * 1. Employees Working Since 1999

SELECT

first\_name,

last\_name,

hire\_date

FROM

employees

WHERE

YEAR(hire\_date) >= 1999

ORDER BY

hire\_date DESC,

first\_name ASC;

For this report I selected all the records from the ‘employees’ but filtered the results to only return those that the year of employment is greater than or equal to 1999. After that I added two sort columns so that the results are first sorted by the ‘hire\_date’ descending order and then by ‘first\_name’ ascending order.

Result:

|  |  |  |
| --- | --- | --- |
| **first\_name** | **last\_name** | **hire\_date** |
| Charles | Johnson | 04/01/2000 |
| Luis | Popp | 07/12/1999 |
| Karen | Colmenares | 10/08/1999 |
| Kimberely | Grant | 24/05/1999 |
| Diana | Lorentz | 07/02/1999 |

* + 1. Average Salary of 'Programmers'

SELECT

AVG(salary) AS average\_salary

FROM

employees INNER JOIN jobs

ON employees.job\_id = jobs.job\_id

WHERE

employees.job\_id = 9; -- 9 = Programmer

For this report I first joined the tables ‘jobs’ and ‘employees’ along the ‘job\_id’ column. I then filtered the data by the ‘job\_id’ with the value 19 which is the value of ‘Programmer’ after that I used the aggregate function ‘AVG’ to calculate the average value in the column ‘salary’ for all the filtered records and displayed it as ‘average\_salary’.

Result:

|  |
| --- |
| **average\_salary** |
| 5760.0 |

* + 1. Employees Managed by Employee ID 114

SELECT

e.employee\_id,

e.first\_name,

e.last\_name,

e.manager\_id,

m.first\_name AS manager\_name

FROM

employees e INNER JOIN employees m

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

WHERE

e.manager\_id = 114; -- 114 = Den

For this report I first joined the table ‘employees’ to itself, creating a recursive join. I also gave each representation of the table a unique name to help reference it during the query. Once I had the table joined I can filter the result by the ‘manager\_id’ required which is 114, the id of Den.

Result:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **employee\_id** | **first\_name** | **last\_name** | **manager\_id** | **manager\_name** |
| 115 | Alexander | Khoo | 114 | Den |
| 116 | Shelli | Baida | 114 | Den |
| 117 | Sigal | Tobias | 114 | Den |
| 118 | Guy | Himuro | 114 | Den |
| 119 | Karen | Colmenares | 114 | Den |

## References

* Learning Guide Unit 1-6  
  <https://my.uopeople.edu/course/view.php?id=7455>
* Sharma, N., Perniu, L., Chong, R. F., Iyer, A., Nandan, C., Mitea, A. C., Nonvinkere, M. & Danubianu, M. (2010). Database fundamentals. IBM Canada.  
  <https://my.uopeople.edu/pluginfile.php/1827130/mod_book/chapter/484065/Database_Fundamentals.pdf>
* Watt, A., & Eng, N. (2014). Database design,  2nd ed. BCcampus, BC Open Textbook Project.   
  <https://opentextbc.ca/dbdesign01/>  
  <https://my.uopeople.edu/pluginfile.php/1827130/mod_book/chapter/484065/Database-Design-2nd-Edition-1560272109.pdf>
* SQL Tutorial – SQL HR Sample Database  
  <https://www.sqltutorial.org/sql-sample-database/>  
  Tables: <https://www.sqltutorial.org/wp-content/uploads/2020/04/sqlserver.txt>  
  Seed Data: <https://www.sqltutorial.org/wp-content/uploads/2020/04/sqlserver-data.txt>