

In this dataset we have data that measures the salaries of individuals based on several factors. Salary information for employees, along with additional details like age, gender, education level, job title, and years of experience.

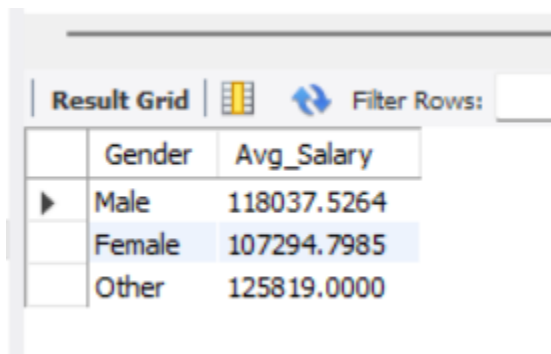
The dataset contains the following columns:

1. **Age:** The age of the employee.
2. **Gender:** The gender of the employee.
3. **Education Level:** The highest education level attained.
4. **Job Title:** The employee's job title.
5. **Years of Experience:** The total number of years of experience the employee has.
6. **Salary:** The annual salary of the employee.

Average salary of each gender:

```
SELECT Gender, AVG(Salary) as Avg_Salary  
FROM salary_database.salary_data_cleaned  
GROUP BY Gender;
```

Output:



The screenshot shows a database interface with a 'Result Grid' tab. It displays a table with two columns: 'Gender' and 'Avg_Salary'. The table contains three rows: 'Male' with an average salary of 118037.5264, 'Female' with 107294.7985, and 'Other' with 125819.0000. The 'Female' row is highlighted with a blue background.

Gender	Avg_Salary
Male	118037.5264
Female	107294.7985
Other	125819.0000



Top 5 Highest Paying Job Titles:

```
SELECT `Job Title`, AVG(Salary) AS Avg_Salary  
FROM salary_database.salary_data_cleaned  
GROUP BY `Job Title`
```

ORDER BY Avg_Salary DESC

LIMIT 5;

Output:

Result Grid   Filter Rows: <input type="text"/>		
	Job Title	Avg_Salary
▶	Chief Technology Officer	250000.0000
	CEO	250000.0000
	Chief Data Officer	220000.0000
	Director of Data Science	200000.0000
	Director	200000.0000

Salary Distribution by Years of Experience

SELECT `Years of Experience`, AVG(Salary) AS Avg_Salary

FROM salary_database.salary_data_cleaned

GROUP BY `Years of Experience`

ORDER BY `Years of Experience` ASC;

Output:

	Years of Experience	Avg_Salary
▶	0	37021.1500
	1	44444.9921
	2	53382.7284
	3	66083.8451
	4	77462.8468
	5	92566.9615
	6	107234.2286
	7	110350.5761
	8	114675.3619
	9	127693.0824
	10	123998.5455
	11	142040.1806
	12	143873.5125
	13	144199.9032
	14	155265.2063
	15	154780.9574
	16	170104.3390

Gender Distribution by Job Title

```
SELECT `Job Title`, Gender, COUNT(*) AS Count
```

```
FROM salary_database.salary_data_cleaned
```

```
GROUP BY `Job Title`, Gender;
```

Output:

Result Grid			
Filter Rows:			
	Job Title	Gender	Count
►	Software Engineer	Male	55
	Data Analyst	Female	21
	Senior Manager	Male	2
	Sales Associate	Female	11
	Director	Male	1
	Marketing Analyst	Male	10
	Product Manager	Female	29
	Sales Manager	Male	11
	Marketing Coordinator	Female	25
	Senior Scientist	Male	2
	Software Developer	Male	13
	HR Manager	Female	2
	Financial Analyst	Male	9
	Project Manager	Female	1
	Customer Service Rep	Male	1
	Operations Manager	Male	20
	Marketing Manager	Female	37
	Senior Engineer	Male	2

Employees Earning Above 100k

```
SELECT *
FROM salary_database.salary_data_cleaned
WHERE Salary >= 100000;
```

Output:

Result Grid						
		Filter Rows:	Export:		Wrap Cell Content:	
	Age	Gender	Education Level	Job Title	Years of Experience	Salary
	31	Male	Bachelor's	Product Manager	8	190000
	30	Male	Master's	Software Engineer	6	170000
	32	Female	Bachelor's	Data Scientist	9	185000
	29	Male	Bachelor's	Data Analyst	7	130000
	35	Female	Bachelor's	Product Manager	12	190000
	27	Male	Bachelor's	Software Engineer	5	150000
	28	Female	Master's	Data Scientist	6	150000
	37	Male	PhD	Product Manager	14	195000
	26	Male	Bachelor's	Data Analyst	3	110000
	30	Female	Bachelor's	Software Engineer	8	180000
	31	Male	Master's	Data Scientist	9	185000
	29	Female	Bachelor's	Data Analyst	7	130000
	32	Male	Bachelor's	Software Engineer	10	190000
	28	Female	Master's	Data Scientist	6	155000
	34	Male	Bachelor's	Software Engineer	12	195000
	30	Male	Bachelor's	Product Manager	8	185000
	29	Female	Bachelor's	Data Analyst	6	120000

Number of Employees by Education Level

```
SELECT `Education Level`, count(*) AS degree_count
FROM salary_database.salary_data_cleaned
GROUP BY `Education Level`
ORDER BY `Education Level`;
```

In this query, I observed that while the data was sorted correctly, there were duplicate categories for certain education levels, specifically for bachelor's and master's degrees. For example, there are entries labeled as both "Bachelor's" and "Bachelor's Degree." Although these represent the same degree level, MySQL treats them as separate categories, resulting in distinct counts for each.

To ensure accurate grouping, these variations need to be standardized into a single category for each degree level.

Output:

Result Grid			Filter Rows:
	Education Level	degree_count	
▶		1	
	Bachelor's	262	
	Bachelor's Degree	506	
	High School	110	
	Master's	122	
	Master's Degree	446	
	PhD	341	

To fix this issue I decided to UPDATE the education level category of for example “Bachelors Degree” to “Bachelors” so the count can be counted only in one distinct option.

I wrote this query to update it:

Standardize Bachelor's Degree variations

UPDATE salary_database.salary_data_cleaned

SET `Education Level` = 'Bachelors'

WHERE `Education Level` IN ('Bachelor\'s Degree', 'Bachelor\'s');

Standardize Master's Degree variations

UPDATE salary_database.salary_data_cleaned

SET `Education Level` = 'Masters'

WHERE `Education Level` IN ('Master\'s Degree', 'Master\'s');

I encountered an error stating: “Error Code: 1175. You are using safe update mode and you tried to update a table without a WHERE that uses a KEY column. To disable safe mode, toggle the option in Preferences -> SQL Editor and reconnect.”

After looking into it, I learned that this is a MySQL safety feature intended to prevent accidental updates on large datasets without a unique identifier in the WHERE clause. This feature helps

avoid unintended changes to large amounts of data. Since my dataset isn't very large, I chose to temporarily disable safe mode and proceeded with the update.

#Run the following command to turn off safe mode for the current session

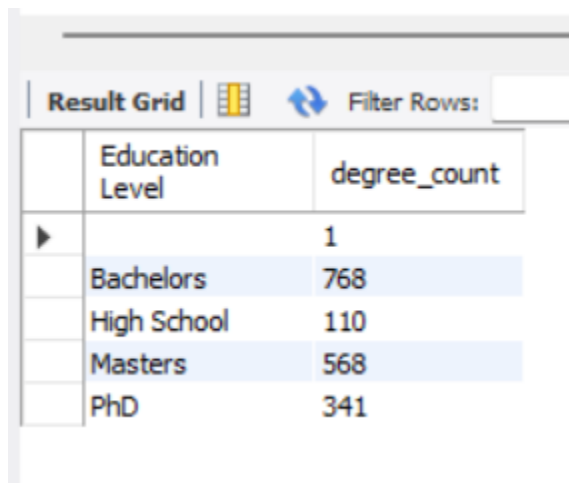
```
SET SQL_SAFE_UPDATES = 0;
```

After I ran it again and re enable the safety feature:

```
SET SQL_SAFE_UPDATES = 1;
```

Afterwards we ran my query again and got the results. Way easier to read and categorize!

Output:



The screenshot shows a SQL query result grid with two columns: 'Education Level' and 'degree_count'. The results are as follows:

Education Level	degree_count
	1
Bachelors	768
High School	110
Masters	568
PhD	341

Average Salary by Education Level:

```
SELECT `Education Level`, AVG(Salary) AS Avg_Salary  
FROM salary_database.salary_data_cleaned  
GROUP BY `Education Level`;
```

Output:

Result Grid			Filter Rows:
	Education Level	Avg_Salary	
▶	Bachelors	91145.5664	
	Masters	127432.8063	
	PhD	160942.5220	
		100000.0000	
	High School	45435.9818	

Total Salary by Job Title

```
SELECT `Job Title`, SUM(Salary) AS `Salary Total`
FROM salary_database.salary_data_cleaned
GROUP BY `Job Title`
ORDER BY `Salary Total` DESC;
```

Output:

Result Grid



Filter Rows:

	Job Title	Salary Total
▶	Software Engineer Manager	21839694
	Full Stack Engineer	15693558
	Senior Software Engineer	14820070
	Senior Project Engineer	14434690
	Data Scientist	13018000
	Software Engineer	8654000
	Back end Developer	8056506
	Product Manager	7510000
	Front end Developer	6750619
	Marketing Manager	6213000
	Data Analyst	6023000
	Director of Marketing	4060000
	Director of HR	3300000
	Financial Manager	3295000
	Content Marketing Manager	2950000
	Operations Manager	2310000
	Research Scientist	2260000