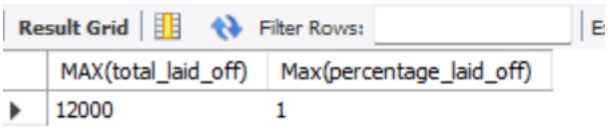
-- Exploratory Data Analysis

Performed various operation on the layoffs data

To find the Maximum layoffs

```
SELECT MAX(total_laid_off), Max(percentage_laid_off)
FROM layoffs_staging2;
```



Result Grid | Filter Rows:

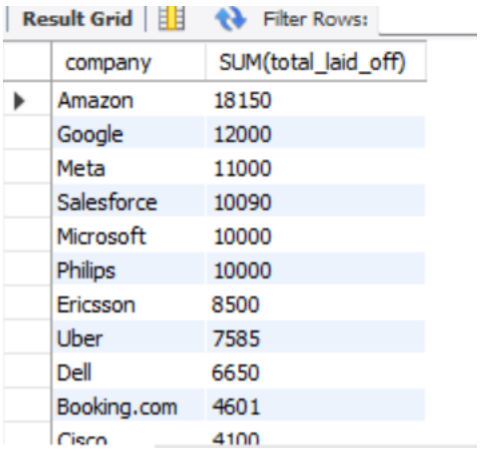
MAX(total_laid_off)	Max(percentage_laid_off)
12000	1

To find the details of companies who raised the maximum funds

```
SELECT *
FROM layoffs_staging2
WHERE percentage_laid_off = 1
ORDER BY funds_raised_millions DESC;
```

To find the company name and number of laid_off by company

```
SELECT company, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company
ORDER BY 2 DESC;
```



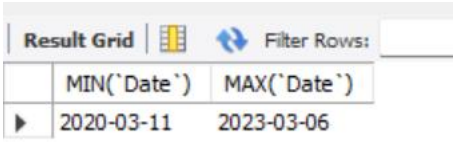
Result Grid | Filter Rows:

company	SUM(total_laid_off)
Amazon	18150
Google	12000
Meta	11000
Salesforce	10090
Microsoft	10000
Philips	10000
Ericsson	8500
Uber	7585
Dell	6650
Booking.com	4601
Cisco	4100

```
SELECT *
FROM layoffs_staging2;
```

Start and end date of the layoffs data

```
SELECT MIN(`Date`),  MAX(`Date`)
FROM layoffs_staging2;
```



Result Grid | Filter Rows:

	MIN(`Date`)	MAX(`Date`)
▶	2020-03-11	2023-03-06

According to the industry, total no. of layoffs

```
SELECT industry, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY industry
ORDER BY 2 DESC;
```

According to the country, total no. of layoffs

```
SELECT country, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY country
ORDER BY 2 DESC;
```

```
SELECT * FROM layoffs_staging2;
```

According to the `date`, total no. of layoffs

```
SELECT `date`, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY `Date`
ORDER BY 1 DESC;
```

According to the company, total no. of layoffs

```
SELECT company, SUM(percentage_laid_off)
FROM layoffs_staging2
GROUP BY company
ORDER BY 2 DESC;
```

Montly layoffs happened

```
SELECT substring(`Date`,1,7) AS `Month`, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY `Month`
ORDER BY `Month`ASC;
```

Used the CTE function

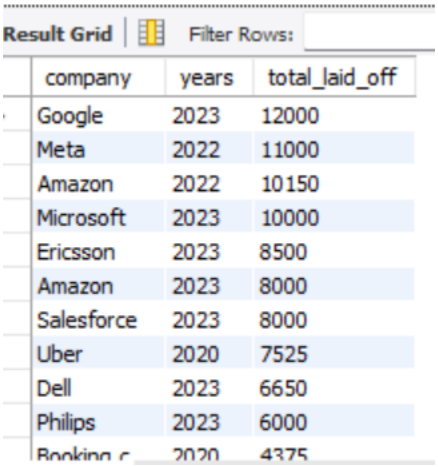
```
WITH Rolling_Total AS
(
SELECT substring(`Date`,1,7) AS `Month`, SUM(total_laid_off) AS total_off
FROM layoffs_staging2
GROUP BY `Month`
ORDER BY `Month`ASC
)
SELECT `Month`, total_off,
SUM(total_off) OVER(ORDER BY `Month`) AS rolling_total
FROM Rolling_Total;
```

```
SELECT company, SUM(total_laid_off)
```

```
FROM layoffs_staging2
GROUP BY company
ORDER BY 2 DESC;
```

Arranged the layoffs according to Year

```
SELECT company,YEAR(`date`), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company,YEAR(`date`)
ORDER BY 3 DESC;
```



Result Grid | Filter Rows:

company	years	total_laid_off
Google	2023	12000
Meta	2022	11000
Amazon	2022	10150
Microsoft	2023	10000
Ericsson	2023	8500
Amazon	2023	8000
Salesforce	2023	8000
Uber	2020	7525
Dell	2023	6650
Philips	2023	6000
Booking.c	2020	4375

```
WITH Company_Year (company, years, total_laid_off) AS
(
SELECT company,YEAR(`date`), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company,YEAR(`date`)
ORDER BY 3 DESC
)
```

```
SELECT *
FROM Company_Year;


Adding a Ranking Column  for understanding the most layoffs in which company and year
WITH Company_Year (company, years, total_laid_off) AS
(
SELECT company,YEAR(`date`), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company,YEAR(`date`)
ORDER BY 3 DESC
), Company_Year_Rank AS
(
SELECT *, DENSE_RANK() OVER(PARTITION BY years ORDER BY total_laid_off DESC) AS RANKING
FROM Company_Year
WHERE years IS NOT NULL
ORDER BY RANKING ASC
)
SELECT *
FROM Company_Year_Rank;
```

Result Grid | Filter Rows: | Export:

company	years	total_laid_off	RANKING
Uber	2020	7525	1
Bytedance	2021	3600	1
Meta	2022	11000	1
Google	2023	12000	1
Booking.com	2020	4375	2
Katerra	2021	2434	2
Amazon	2022	10150	2
Microsoft	2023	10000	2
Groupon	2020	2800	3
Zillow	2021	2000	3
Cisco	2022	4100	3
Ericsson	2023	8500	3
Swiggy	2020	2250	4
Instacart	2021	1877	4

Finding the top 5 companies having large number of layoffs

```
WITH Company_Year (company, years, total_laid_off) AS
(
SELECT company,YEAR(`date`), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company,YEAR(`date`)

), Company_Year_Rank AS
(
SELECT *, DENSE_RANK() OVER(PARTITION BY years ORDER BY total_laid_off DESC) AS RANKING
FROM Company_Year
WHERE years IS NOT NULL
)
SELECT *
FROM Company_Year_Rank
WHERE Ranking <= 5;
```

Result Grid | Filter Rows: | Export:

company	years	total_laid_off	RANKING
Uber	2020	7525	1
Booking.com	2020	4375	2
Groupon	2020	2800	3
Swiggy	2020	2250	4
Airbnb	2020	1900	5
Bytedance	2021	3600	1
Katerra	2021	2434	2
Zillow	2021	2000	3
Instacart	2021	1877	4
WhiteHat Jr	2021	1800	5
Meta	2022	11000	1
Amazon	2022	10150	2
Cisco	2022	4100	3
Peloton	2022	4084	4
Carvana	2022	4000	5
Philips	2022	4000	5
Google	2023	12000	1
Microsoft	2023	10000	2
Ericsson	2023	8500	3