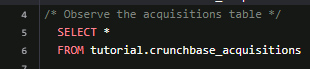
**Crunchbase data**

Crunchbase data is a crowdsourced index of startups, founders, investors, and activities. The specific data was collected February 5th, 2014. This exploration consists of multiple joins and clauses to look at various data on investors for the startups and which are acquired and are not. Lastly, UNIONs are used between 2 datasets.





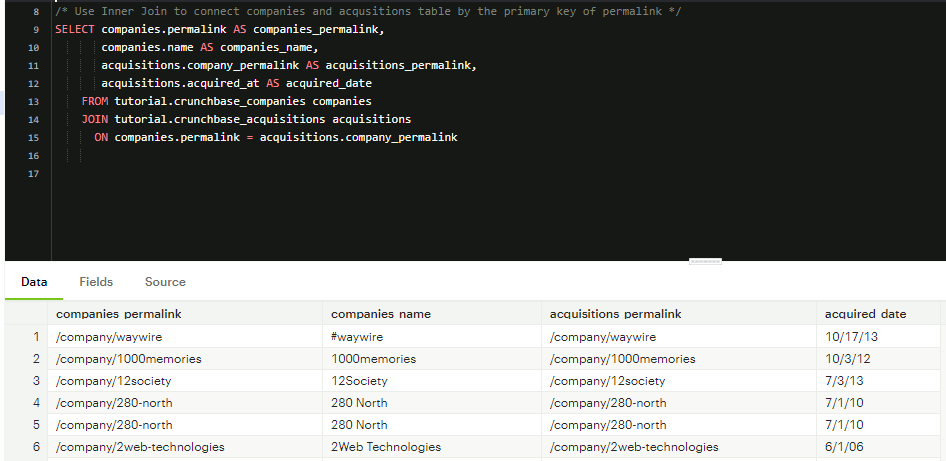
- There are multiple columns, some to observe are: permalink it is a unique identifier for each row, we also see the web address.





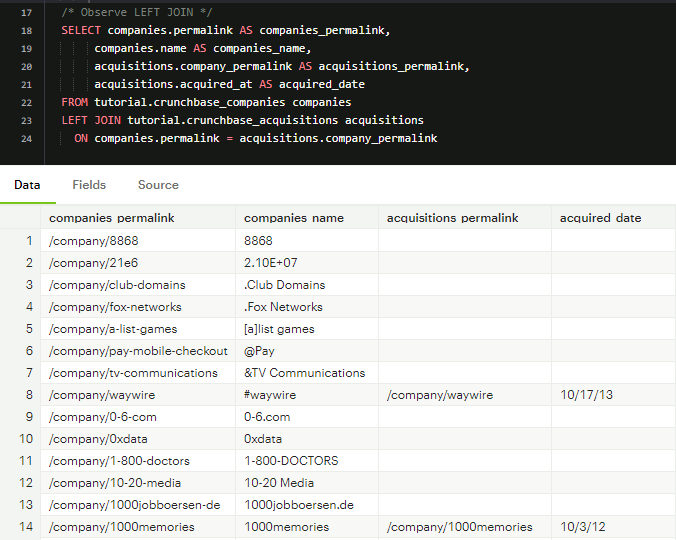
- This second table consists of acquisitions. This table has a column named company permalink, this table maps to the permalink companies table. We can join both of these tables together to gain additional information about the companies being acquired.

**Inner Join**

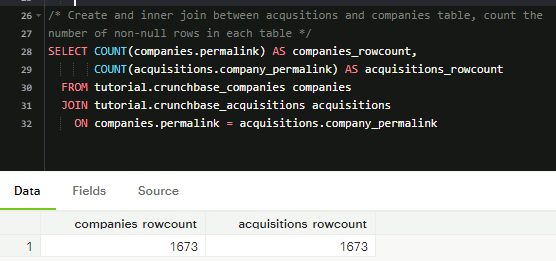


* Connection using Inner Join is made between the companies table and acquisitions table. We use the primary key of permalink. We observe 280 North is shown twice, it has two entries in the tutorial.crunchbase\_acquisitons table

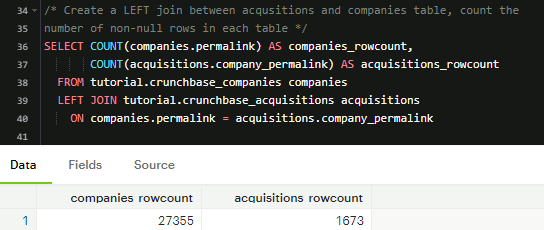
**LEFT JOIN**



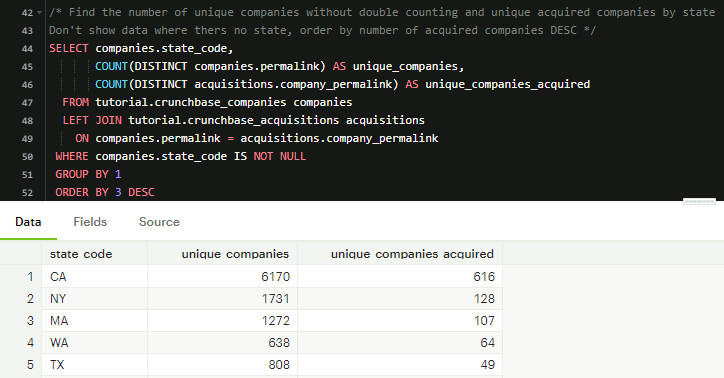
- When using LEFT JOIN, we observe that the first two companies that were in the INNER JOIN results, #waywire and 1000memories, are now in lower positions and now the data itself is surrounded by null values in the acquisitions\_permalink and acquisitions\_date fields. We observe null values because when you use a LEFT JOIN, the command returns ALL rows in the table regardless if there is a match or not in the table.



- Here we performed an INNER JOIN, and used COUNT to count the number of non-null rows in each table. This gives us an insight on the data for non-nulls. Since we used INNER JOIN, it will show matching values as its everything inside the join

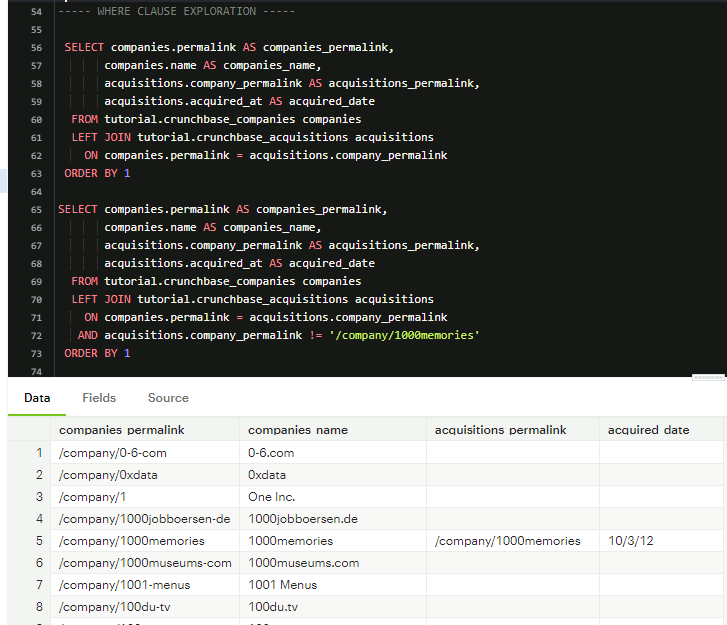


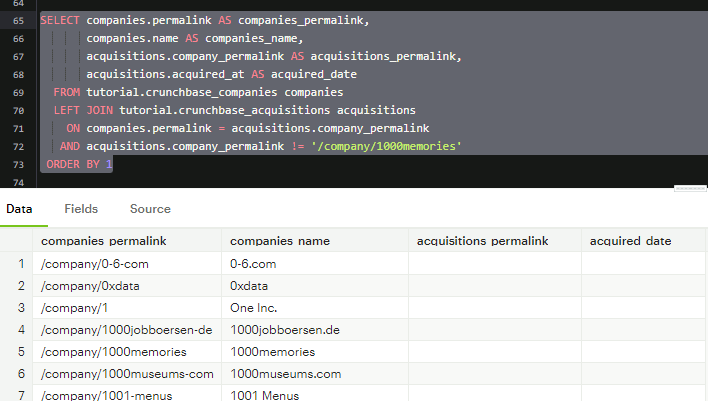
- Here I tested the same query but with LEFT JOIN, now we can observe the non-null values for companies are larger than acquisitions. This is because it pics everything from inside the table and to the left of it.



- Here we are observing the state code, unique companies and unique companies acquired. This was done by doing a count of unique companies and unique acquired companies by state. We do not want to observe NULL values so the WHERE clause was used where state code is NOT NULL, the grouping was done by statecode and ordered by unique companies acquired in DESC order. We see CA and NY have the highest unique startup companies and had the most acquired.

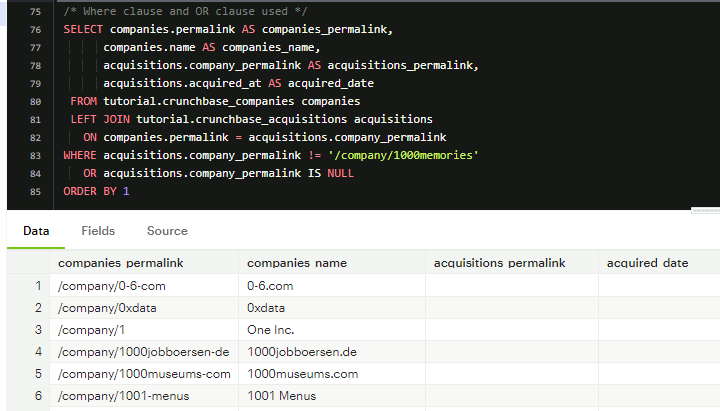
**WHERE CLAUSE EXPLORATION**

****

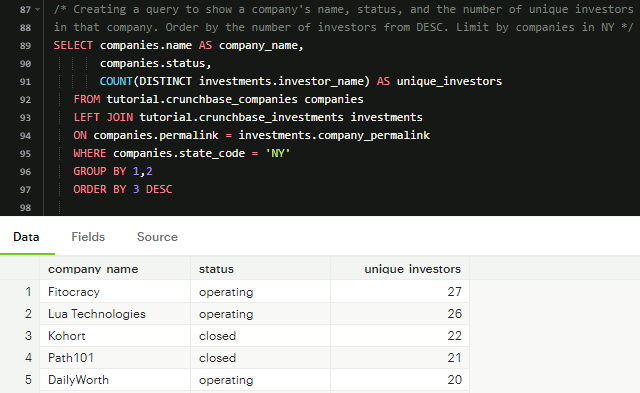
****

- Comparing both of these queries, we see that in the second query everything in the tutorial.crunchbase\_acquisitions table was joined on except for the row which company\_permalink is /company/1000memories. The AND statement is evaluated before the join statement, we notice that 1000memories permalink is still showing in the column that pulls from the other table

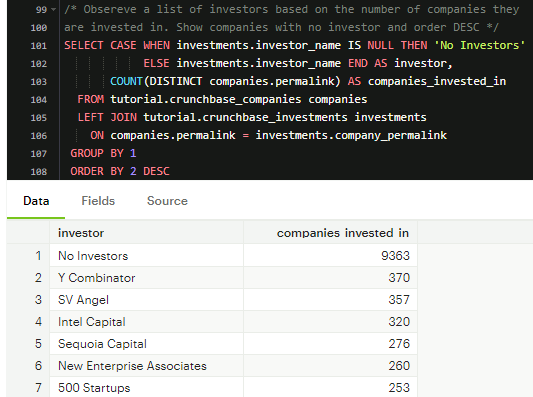
**WHERE CLAUSE**



* Running this query, the WHERE and OR clause are used. Here we see that 1000memories line is not returned



- This query is created to explore the company's name, its status, and the number of unique investors in a company. COUNT DISTINCT clause gets used with the investor name to find a unique investor. We use a LEFT JOIN to join the investments table followed by grouping by company name and status and sorting by highest unique investors to lowest. We observe that Fitocracy is an operating company with 27 investors.

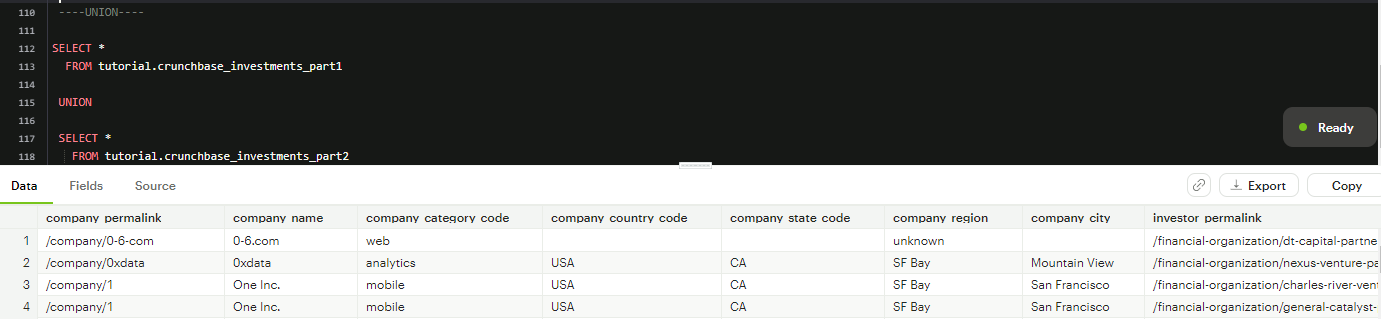


- This query shows a list of investors based on the number of companies invested by investors. CASE and ELSE statements are used to identify No Investors and Investors. Followed by the COUNT DISTINCT clause to count the primary key of companies permalink.

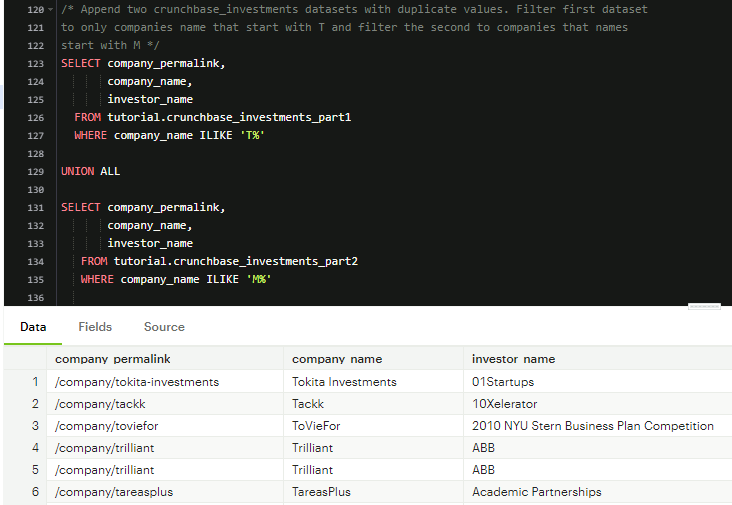
**UNION**

- The UNION operator allows you to stack one dataset on top of another. Lets you write two separate SELECT statements and the results would display in the same table as the results from the other statement. UNION appends distinct values only. If any identical rows are present, the first table will drop them.

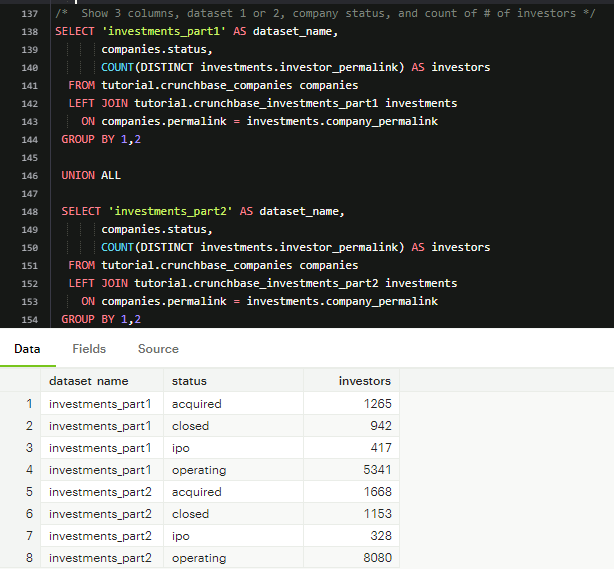
UNION ALL can be used to append values from second table



In SQL for appending data, both tables have to have the same number of columns and columns must have the same data types in the same order as the first table.



- Here we append two of the crunchbase\_investments datasets and are filtering by T and M for the starting letter. We notice that the company name starts with the letter T.



- This query shows 3 columns, first one indicates investment part, second column shows the company status and the third column is the number of investors. We look at this to see the comparison of data from both tables.

**Comparison operators with joins**



* Here we use > to join only the investments that had occurred more than 5 years after each company's founding year. We see the different companies under permalink with names and status if the company is operating or has been acquired.