

SQL & Python for Analytics

USC Annenberg Digital Lounge
Week 1: Intro to SQL

**** → Sign up at: bit.ly/usc-hex-workspace-2024 ****

Today's session

This is going to be interactive, so please be ready to try out the exercises as we go!

We are going to be using a tool called Hex, which is a tool for using SQL and Python for doing data analysis and visualization.

Shout-out to Hex for giving us their Professional Plan for free!

Go to bit.ly/usc-hex-workspace-2024 and sign up with your USC email address.

Workshop overview

- Week 1: Intro to SQL
- Week 2: Data Modeling with SQL
- Week 3: Intro to Python
- Week 4: Analyzing Data with Python
- Week 5: Analytics & Visualization with SQL and Python

What is SQL?

Structured Query Language

- A tool for interacting with a database

A **database** is a structured collection of related data

- The simplest database is... a spreadsheet!

Databases are used by:

- Retailers
- Social Media
- Apps and websites
- And more!

Who uses SQL?

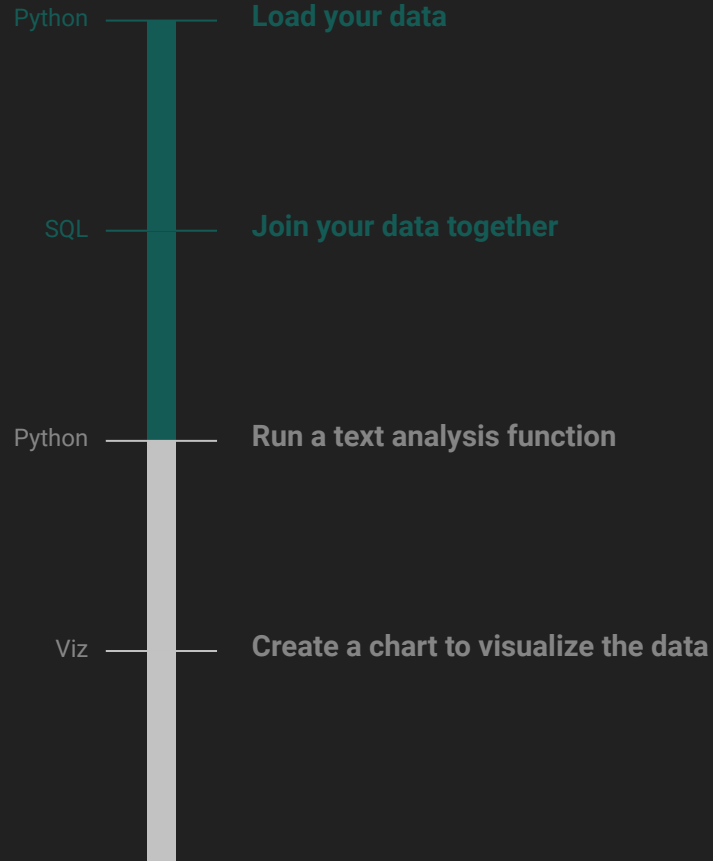
- Technical roles
 - Data analyst
 - Marketing analyst
 - Data scientist
- Non-technical roles
 - Any role at a tech company or startup
 - Media planning
 - Marketing
 - Customer success

What are SQL queries used for?

Find data that meets criteria	Which episodes have the word “Baby” in the title?
Summarize data	How many episodes are in each season of this show? What is the average rating for this show?
Manipulate data	Create a new column that says “Good” if the episode’s rating is >8 and “Bad” otherwise
Group data	To summarize by subgroups
Order data	Sort by episode date or length or rating
Combine data	Do any of the above using data from multiple datasets

What is Hex?

- Hex is a SQL and Python notebook tool
- It works as a series of 'cells' that link together
- Here's an example of some cells:



Activity

1. Go to Projects, next to **Fall 2024 Week 1** hit '...' then Duplicate
2. Load your datasets
 - a. We are going to load them with Python, we'll discuss how it works next week
 - b. Your first cell should look like this – click into it and hit Run in the upper right, or Command-Enter

Fall 2024 Week 1: Intro to SQL

Copy this template to start for Week 1!

   Template

Code 1

```
1 import pandas as pd
2 transcripts_data = pd.read_csv('https://query.data.world/s/2svp7yxiggyn4eui6wvklptslqctle?dws=00000')
3 ratings_data = pd.read_csv("SeinfeldRatingsData.xls")
```



↳ pd transcripts_data ratings_data

How to write a query

SELECT (columns, or a function of columns, or * for all)

FROM (table)

JOIN (other table)

WHERE (conditions)

GROUP BY (columns)

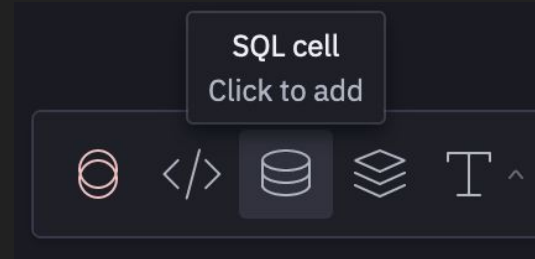
ORDER BY (which column)

LIMIT (if you only want a certain number of results)

Sometimes those words are written in all caps by convention, but it is not required

Write your first query

- Add a new SQL cell to your Hex project



- Write a query to get the first 10 rows of your dataset:

```
SELECT * FROM (table name) LIMIT 10
```

- To run your query, click Run on that cell, or hit Command-Enter
- Add another new SQL cell
- Write a query to get the number of rows of your dataset:

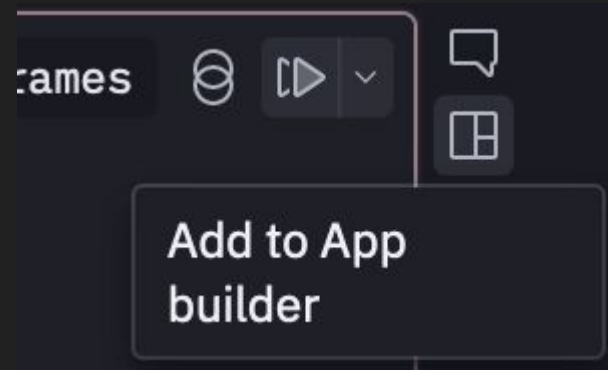
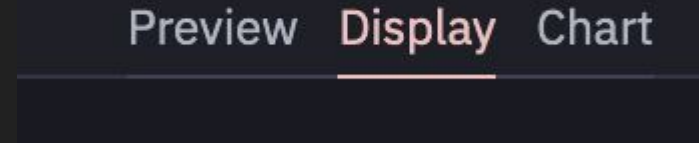
```
SELECT count(*) FROM df
```

Navigating Hex

Once you run a query, you can view the results as:

- Preview (limited view)
- Display (fully functional table view)
- Chart (quick viz)

Once you have a Display or Chart, you can click Add to App builder to show it in the App view (final product for your end users)



Constructing a query from a question

What characters in Seinfeld speak the most often?

```
SELECT character FROM transcripts_data
```

But a lot of the characters are repeated...

```
SELECT DISTINCT character FROM transcripts_data
```

Constructing a query from a question, continued

But we still need to add how many times each speaks...

SELECT DISTINCT character, count(*)

FROM transcripts_data

GROUP BY character

But the results aren't in order...

ORDER BY 2 DESC

SQL 6

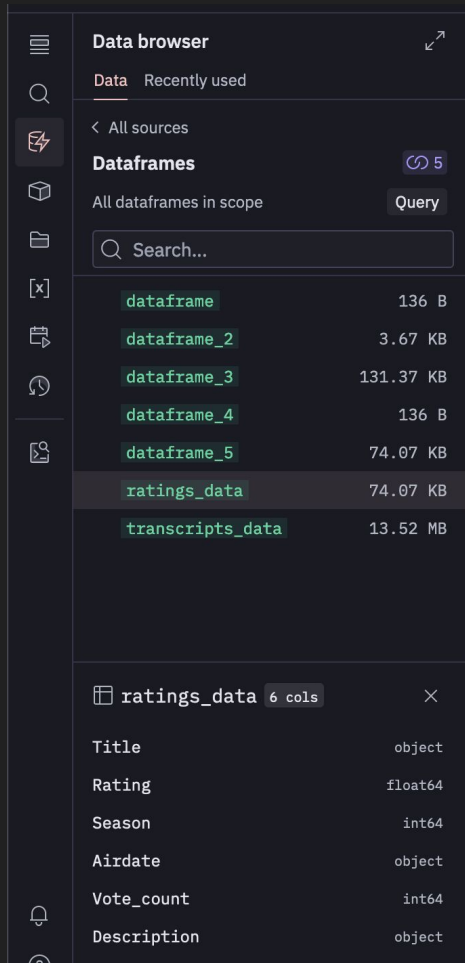
```
1  select distinct character,  
2  count(*)  
3  from transcripts_data  
4  group by 1  
5  order by 2 desc
```

Navigating Hex

How do I know what columns are available to select from?

Go to Data Browser → choose a dataframe

See all the columns listed below



The screenshot shows the Hex Data Browser interface. At the top, there's a 'Data browser' header with a search icon and a 'Data' tab. Below the header, there's a navigation bar with 'All sources' and 'Dataframes' (5). A search bar is present. The main area displays a list of dataframes with their names and sizes:

Dataframe Name	Size
dataframe	136 B
dataframe_2	3.67 KB
dataframe_3	131.37 KB
dataframe_4	136 B
dataframe_5	74.07 KB
ratings_data	74.07 KB
transcripts_data	13.52 MB

Below the list, the 'ratings_data' dataframe is selected, showing its schema with 6 columns:

Column Name	Column Type
Title	object
Rating	float64
Season	int64
Airdate	object
Vote_count	int64
Description	object

Activity

1. Write a query to answer this question: What season has the highest average episode rating?

Remember, start very simply, then add on to your query!

*You'll need to use **average(column)** instead of **count(column)***

2. In another cell, write a query to answer this question: What are the names of all the episodes in the highest-rated season?

*You'll need to use a **WHERE column = value***

Using Hex Magic for an AI assist

One of the “new cell” options is Hex Magic, which is an AI assistant for creating SQL, python, or visualizations

The way I'd recommend using this is as a first draft, that you'll then revise to get the result you want

You'll still need to understand how SQL and Python work, to be able to use this effectively!

Here are two examples:

Hex Magic Example #1

I gave this prompt:

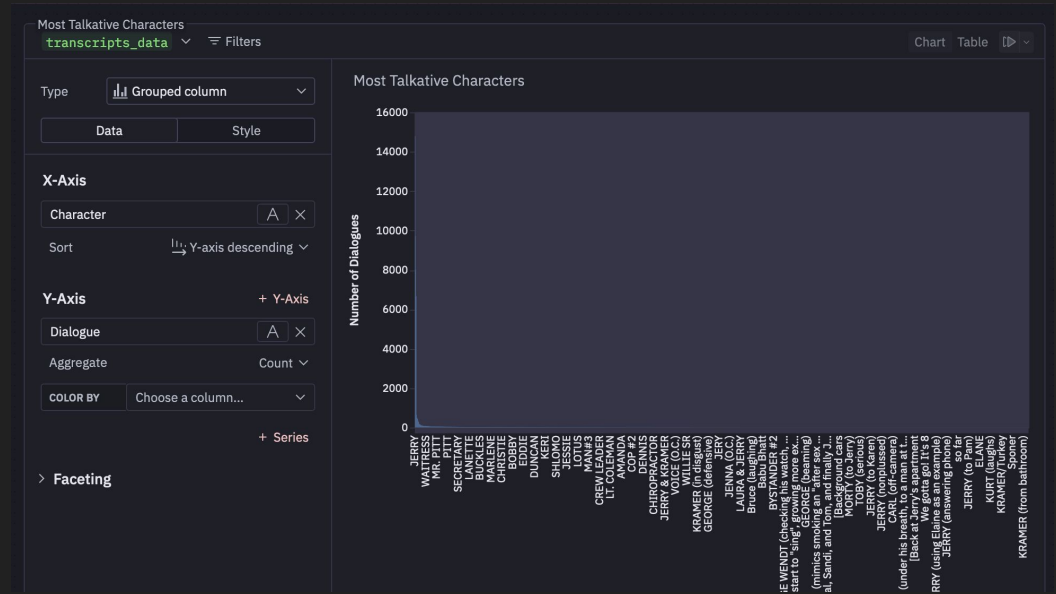


which characters speak the most?

Use @ for tables and dataframes. [Learn how Magic works](#)

And Magic created this table:

Kinda answers my question, but needs some tweaks



Hex Magic Example #2

I gave the prompt:

And it gave me a query that's not too far off... for a database that doesn't exist.

How could I use this as a starting point?

Edit this SQL myself!

write a sql query for which character talks the most

Top Character Lines

0 r

```
1 SELECT
2     CHARACTER,
3     COUNT(*) AS LINE_COUNT
4 FROM DEMOS.MOVIE_DATA
5 GROUP BY CHARACTER
6 ORDER BY LINE_COUNT DESC
7 LIMIT 1;
```

Filters



Your query could not be executed

We received the following error when executing this query:

Hide error details

Exception: Database returned error while executing query
SQL compilation error:
Schema 'ANALYTICS.DEMOS' does not exist or not authorized.
traceId: fb47e84a064e4dceae2ba67ef5e8ef33

↳ character_with_most_lines

Hex Magic Example #2, fixed!

I fixed the table name, updated the data source selector, and hit run!

Looking at the results... that sounds right!

write a sql query for which character talks the most

Top Character Lines

0 r

```
1 SELECT
2     CHARACTER,
3     COUNT(*) AS LINE_COUNT
4 FROM DEMO.MOVIE_DATA
5 GROUP BY CHARACTER
6 ORDER BY LINE_COUNT DESC
7 LIMIT 1;
```

Top Character Lines

```
1 SELECT
2     CHARACTER,
3     COUNT(*) AS LINE_COUNT
4 FROM transcripts_data
5 GROUP BY CHARACTER
6 ORDER BY LINE_COUNT DESC
7 LIMIT 1;
```

Filters

	Character	LINE_COUNT
0	JERRY	14786

Previewing up to 10 rows

Joining data

If you have 2 datasets that have columns with the same value, you can join them together!

For example, if we have season and episode number in 2 datasets, we can join those datasets together using those numbers

Looks like our Ratings dataset is missing episode numbers though... how can we add those, so that we can join our datasets?

Adding columns

- You can select all columns with *
- You can also add a comma and then use functions to add more formulas
 - Many of these are very similar to Excel formulas!
- We're going to walk through an example of a more complex SQL function here:

```
1 SELECT
2     *,
3     ROW_NUMBER() OVER (PARTITION BY season) AS episode_number
4 FROM ratings_data
5 order by season
```

Back to joining data

If you have 2 datasets that have columns with the same value, you can join them!

So now our Ratings dataset has season and episode numbers (one row per episode), and our Transcripts dataset has many rows per episode.

To do a 1:1 join, we want to first summarize our Transcripts data, so that it has one row per episode

```
with transcripts as (  
  select season,  
         episodeNo as episode_number,  
         count(case when Character = 'ELAINE' then Dialogue end) as num_elaine_lines  
  from transcripts_data  
  group by 1, 2  
)  
  
select *  
from season_episode_ranking  
left join transcripts using (season, episode_number)
```

Activity

1. Add 3 additional columns to our summarized transcript data to show the number of lines by JERRY, GEORGE, and KRAMER
2. Add a WHEN clause to your query, to filter the results down to only Season 3

Keep learning!

Upcoming sessions:

- Week 2: Data Modeling with SQL
- Week 3: Intro to Python
- Week 4: Analyzing Data with Python
- Week 5: Analytics & Visualization with SQL and Python

Good tutorials at:

- learn.hex.tech
- Learn SQL via a Murder Mystery: <https://mystery.knightlab.com/>
- sqlbolt.com for a more standard tutorial