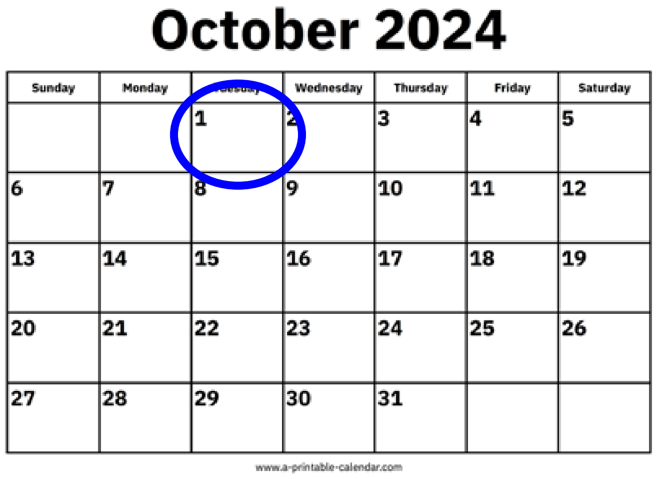
CS 310 : Scalable Software Architectures

Class session on Tuesday, October 1st



October 2024

Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

www.a-printable-calendar.com

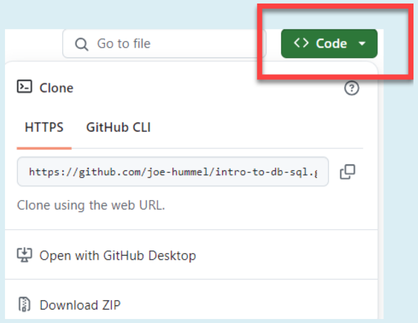
Notes:

- ***Focus this week****:*
  - *Relational databases*
- ***Class sessions *are* being recorded this week***
  - *Will be available under Panopto on Canvas*
- ***Project 01*** *due Wednesday Oct 9th @ 11:59pm*
  - *Build a simple photo app using AWS*
  - *Part 01 has been released, configuration of AWS*
  - *Part 02 to be released, client-side programming*
- ***Office hours*** *started Monday*
- *Optional SQL* ***homework*** *will be posted*

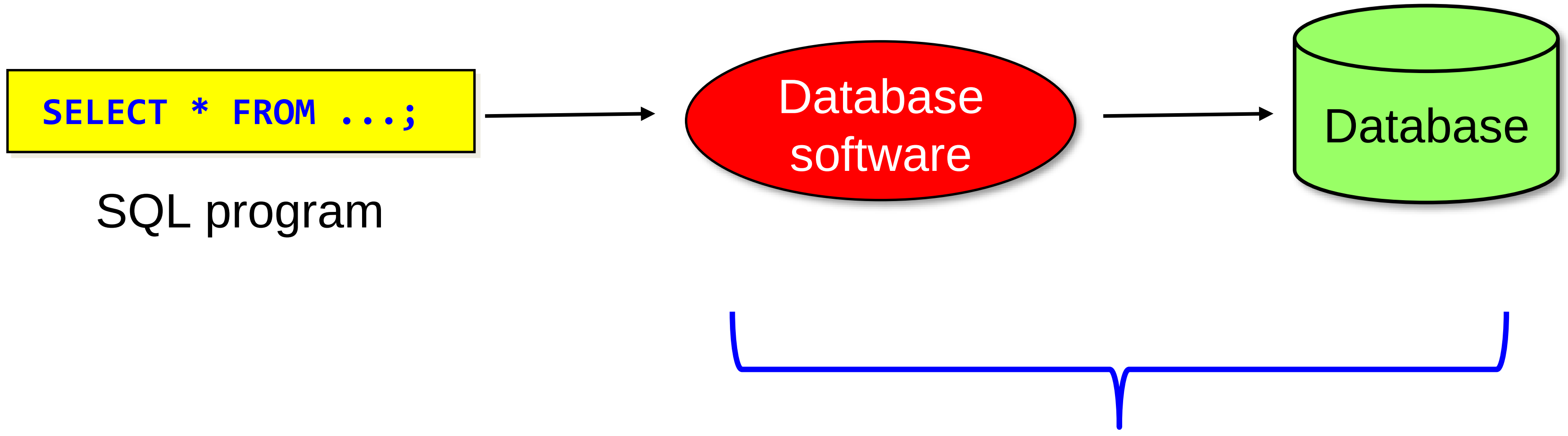
Northwestern University

Goals for today

- **Summarize key points of lecture**
- **Write SQL**
  - *Using sqlite3, a local file-based DBMS (e.g. used in mobile apps)*
  - *Using MySQL running in AWS*
- **Work with Docker**
  - *You need Docker Desktop installed*
  - *Download files from GitHub:*
    - https://github.com/joe-hummel/intro-to-db-sql
    - Clone repo or download ZIP



SQL + DBMS



Database Management System (DBMS)

(MySql, Oracle, SQLite, SQL Server, ...)

SQLite is free, local , file based database, useful for local small devices like mobile.

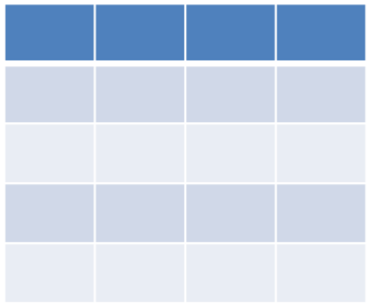
Select

- For retrieving data from a database
- General format:

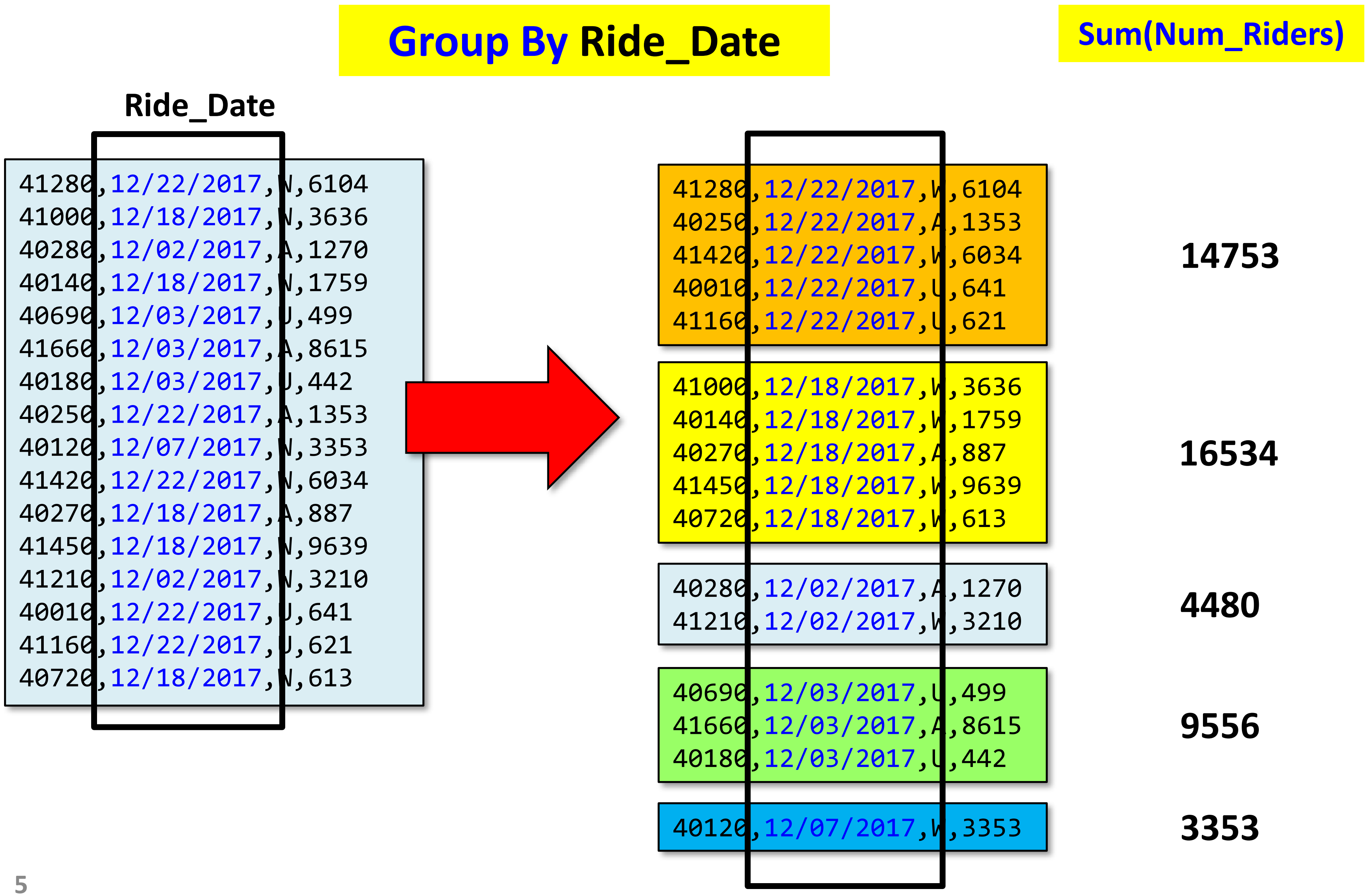
```
SELECT <<the data you want>>
FROM   <<table>>
[ JOIN <<table> ON <<condition>> ]
[ WHERE     <<condition(s)>> ]
[ GROUP BY  <<one or more fields>> ]
[ HAVING    <<condition(s)>> ]
[ ORDER BY  <<one or more fields>> ]
;
```

optional

The **result** is *always* a table



Group by



Group By Ride_Date

Sum(Num_Riders)

Ride_Date

```
41280,12/22/2017,W,6104
41000,12/18/2017,W,3636
40280,12/02/2017,A,1270
40140,12/18/2017,W,1759
40690,12/03/2017,U,499
41660,12/03/2017,A,8615
40180,12/03/2017,U,442
40250,12/22/2017,A,1353
40120,12/07/2017,W,3353
41420,12/22/2017,W,6034
40270,12/18/2017,A,887
41450,12/18/2017,W,9639
41210,12/02/2017,W,3210
40010,12/22/2017,U,641
41160,12/22/2017,U,621
40720,12/18/2017,W,613
```

```
41280,12/22/2017,W,6104
40250,12/22/2017,A,1353
41420,12/22/2017,W,6034
40010,12/22/2017,U,641
41160,12/22/2017,U,621
```

14753

```
41000,12/18/2017,W,3636
40140,12/18/2017,W,1759
40270,12/18/2017,A,887
41450,12/18/2017,W,9639
40720,12/18/2017,W,613
```

16534

```
40280,12/02/2017,A,1270
41210,12/02/2017,W,3210
```

4480

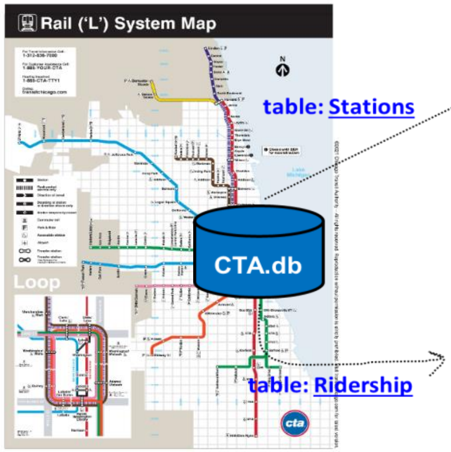
```
40690,12/03/2017,U,499
41660,12/03/2017,A,8615
40180,12/03/2017,U,442
```

9556

```
40120,12/07/2017,W,3353
```

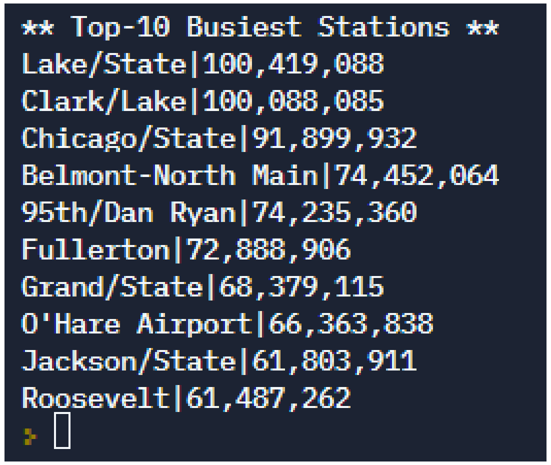
3353

Join



Station_ID	Station_Name
40010	*Austin-Forest Park*
40020	*Harlem-Lake*
40030	*Pulaski-Lake*
…	…

Station_ID	Ride_Date	Type_of_Day	Num_Riders
41280	2017-12-22 00:00:00.000	W	6104
40010	2017-12-28 00:00:00.000	W	1155
40280	2017-12-02 00:00:00.000	A	1270
40030	2017-12-24 00.00.00.000	U	595
…	…	…	…

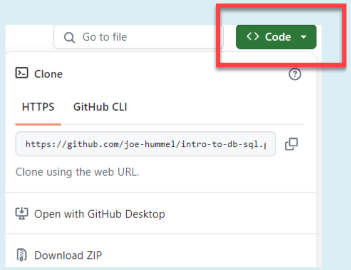


```
** Top-10 Busiest Stations **
Lake/State|100,419,088
Clark/Lake|100,088,085
Chicago/State|91,899,932
Belmont-North Main|74,452,064
95th/Dan Ryan|74,235,360
Fullerton|72,888,906
Grand/State|68,379,115
O'Hare Airport|66,363,838
Jackson/State|61,803,911
Roosevelt|61,487,262
>
```

```sql
select "** Top-10 Busiest Stations **";

Select Station_Name, Sum(Num_Riders)
From   Stations
Join   Ridership On Stations.Station_ID = Ridership.Station_ID
Group By Stations.Station_ID
Order By Sum(Num_Riders) DESC
Limit 10;
```

Getting the necessary software

1. Download files you need for today
   - https://github.com/joe-hummel/intro-to-db-sql
   - Clone repo or download ZIP



2. Make sure Docker Desktop is running
3. Build Docker image and run container:

Linux/Mac/Windows WSL:

1) Open terminal, navigate to folder
2) `chmod 755 *.bash`
3) `./docker-build.bash`
4) `./docker-run.bash`

Windows:

1) Open Powershell, navigate to folder
2) `.\docker-build.bat`
3) `.\docker-run.bat`

```
hummel> ./docker-run.bash
docker> ls
Dockerfile        datatier.py        docker-build.bat   docker-run.bat     main.sql
cta.db            docker-build.bash  docker-run.bash    main.py            movielens.db
docker>
```

Common docker errors

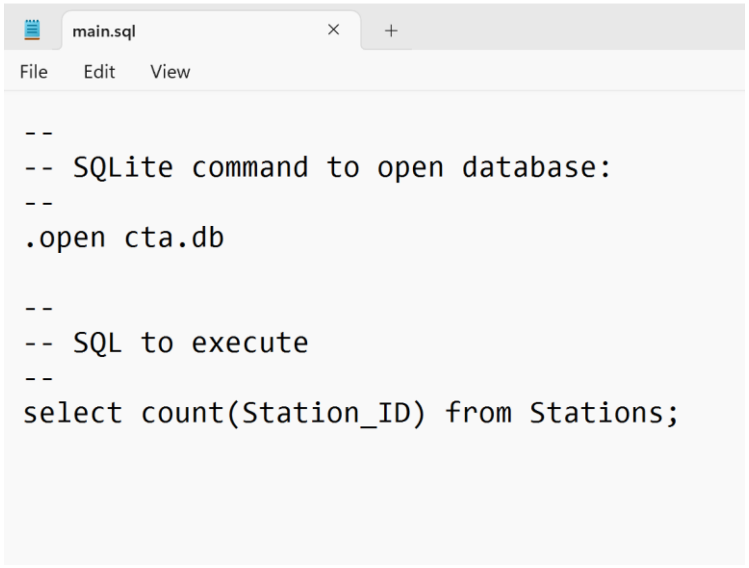
1. **"docker" command not found**
   - *Uninstall and reinstall Docker Desktop*

2. **When you try to build, you are not authorized**
   - *`docker login -u docker-username`*

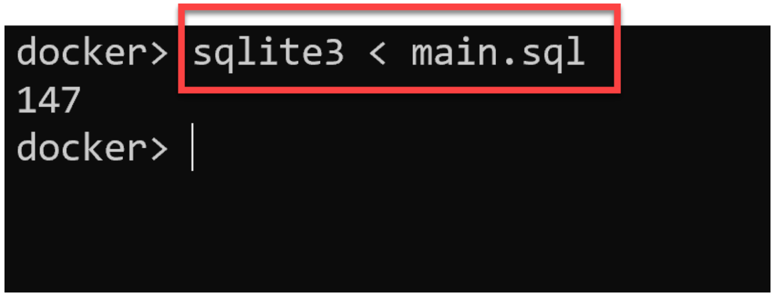
3. **When you try to run, you get errors like "bash: $\r: command not found"**
   1. *If you see the **docker**> prompt, type **exit***
   2. `((Get-Content .bashrc) -join "`n") + "`n" | Set-Content -NoNewLine .bashrc`

Working with sqlite3

1. Open "main.sql" in a text editor
2. Write query and save changes
3. Run via docker container:



```
--
-- SQLite command to open database:
--
.open cta.db

--
-- SQL to execute
--
select count(Station_ID) from Stations;
```



```
docker> sqlite3 < main.sql
147
docker>
```

Exercise

- **What is the yearly ridership for the "Noyes" station?**
  - *Most DBMS have Year( ) function, sqlite does not*

```
SELECT Station_Name,
       strftime('%Y', Ride_Date) as Year,
       Sum(Num_Riders)
FROM     ?  Ridership
INNER JOIN   ?   Station on Ridership.Station_ID = Station.Station_ID
WHERE    ?   Station_Name = ‘Noyes’
GROUP BY   ?   Year
ORDER BY   ?   Year ASC
;
```

```
docker> sqlite3 < main.sql
Noyes|2016|277442
Noyes|2017|282461
Noyes|2018|282356
Noyes|2019|276037
Noyes|2020|85834
Noyes|2021|107387
docker>
```

Stations

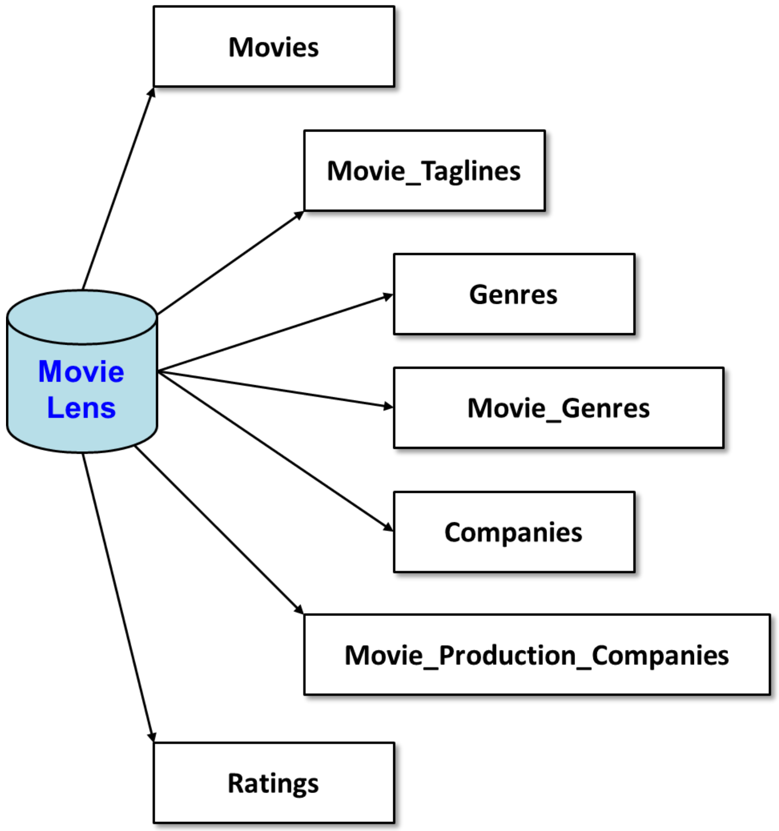
Station_ID	Station_Name
40010	*Austin-Forest Park*
40020	*Harlem-Lake*
40030	*Pulaski-Lake*
…	…

Station_ID	Ride_Date	Type_of_Day	Num_Riders
41280	2017-12-22 00:00:00.000	W	6104
40010	2017-12-28 00:00:00.000	W	1155
40280	2017-12-02 00:00:00.000	A	1270
40030	2017-12-24 00.00.00.000	U	595
…	…	…	…

Ridership

MovieLens database

- **MovieLens**
  - *https://movielens.org/*



Exercises

- How many movies are there? [45,431]
- How many ratings are there? [100,004]
- What is the movie id for 'The Matrix'? [603]
- Which movie titles contain 'matrix'? [there are 7]

SELECT * from Movies where title like '%matrix%';

Movies

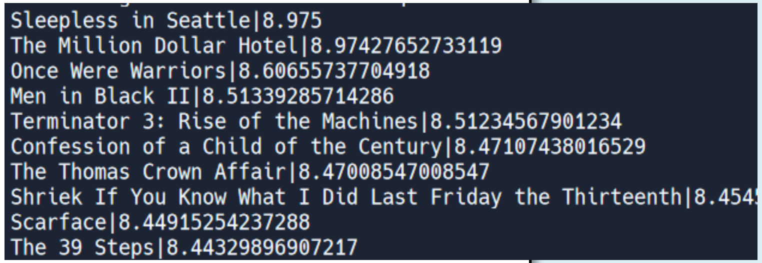
Movie_ID	Title	Release_Date	Runtime	Original_Language	Budget	Revenue
603	The Matrix	1999-03-30 00:00:00.000	136	en	63000000	463517383
862	Toy Story	1995-10-30 00:00:00.000	81	en	30000000	373554033

Ratings

Movie_ID	Rating
605	8
603	6
605	10
605	6

Exercise

```
--
-- Retrieve the top-10 movies ranked by average rating;
-- retrieve the title and average rating. Consider only
-- movies with more than 100 reviews.
--

SELECT ?
FROM ?
INNER JOIN ?
GROUP BY ?
HAVING ?
ORDER BY ?
LIMIT 10;
```



```
Sleepless in Seattle|8.975
The Million Dollar Hotel|8.97427652733119
Once Were Warriors|8.60655737704918
Men in Black II|8.51339285714286
Terminator 3: Rise of the Machines|8.51234567901234
Confession of a Child of the Century|8.47107438016529
The Thomas Crown Affair|8.47008547008547
Shriek If You Know What I Did Last Friday the Thirteenth|8.454
Scarface|8.44915254237288
The 39 Steps|8.44329896907217
```

Movies

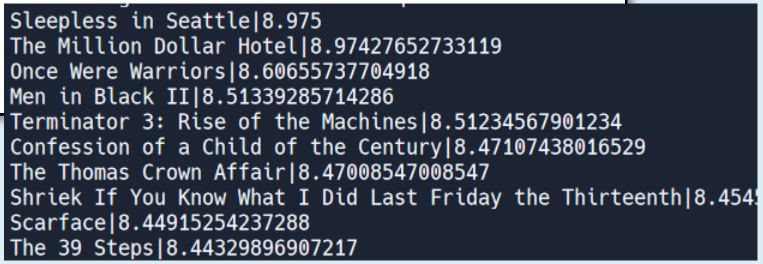
Movie_ID	Title	Release_Date	Runtime	Original_Language	Budget	Revenue
603	The Matrix	1999-03-30 00:00:00.000	136	en	63000000	463517383
862	Toy Story	1995-10-30 00:00:00.000	81	en	30000000	373554033

Ratings

Movie_ID	Rating
605	8
603	6
605	10
605	6

Solution

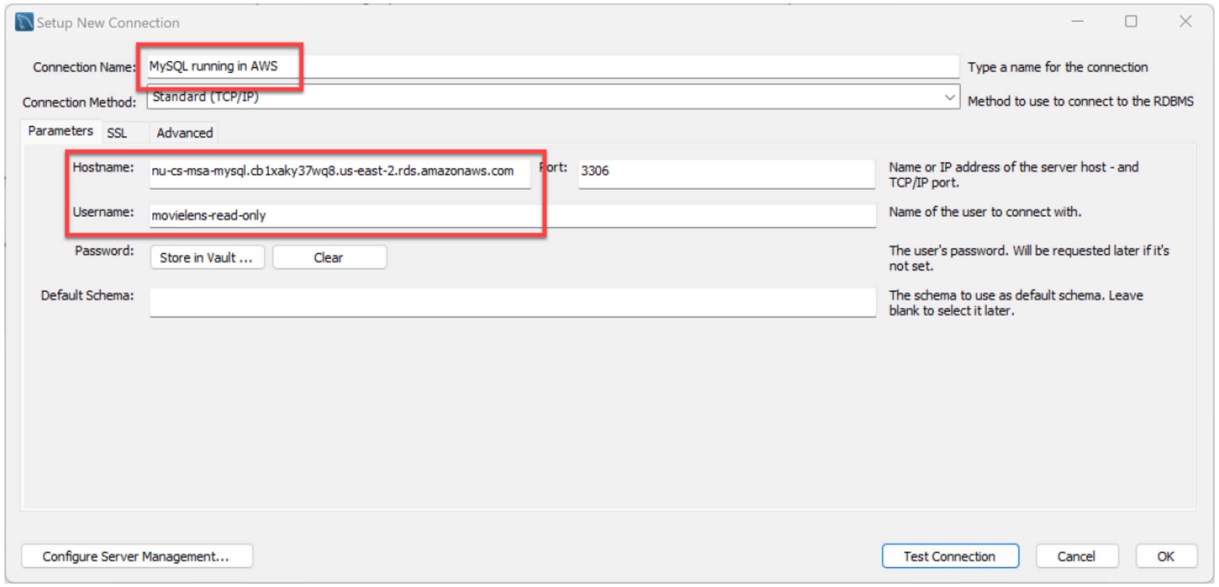
```sql
--
-- Retrieve the top-10 movies ranked by average rating;
-- retrieve the title and average rating. Consider only
-- movies with more than 100 reviews.
--

SELECT Title, avg(Rating) as AvgRating
FROM Movies
INNER JOIN Ratings ON Movies.Movie_ID = Ratings.Movie_ID
GROUP BY Ratings.Movie_ID
HAVING Count(Rating) > 100
ORDER BY AvgRating DESC, Title ASC
LIMIT 10;
```



```
Sleepless in Seattle|8.975
The Million Dollar Hotel|8.97427652733119
Once Were Warriors|8.60655737704918
Men in Black II|8.51339285714286
Terminator 3: Rise of the Machines|8.51234567901234
Confession of a Child of the Century|8.47107438016529
The Thomas Crown Affair|8.47008547008547
Shriek If You Know What I Did Last Friday the Thirteenth|8.454
Scarface|8.44915254237288
The 39 Steps|8.44329896907217
```

Demo: My SQL Workbench

- **In the projects we're going to work with MySQL running in AWS**
  - *Need different software to connect to DB server*

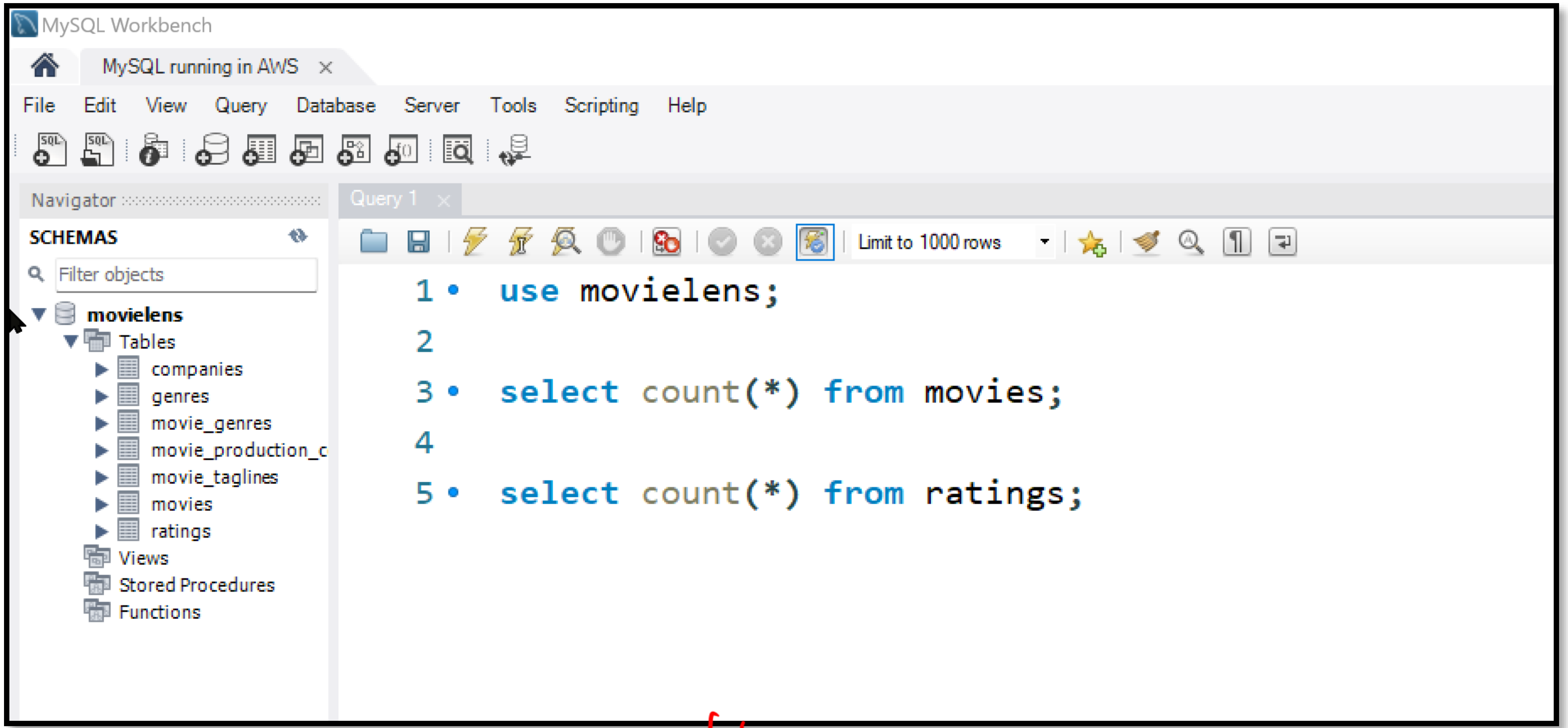


Setup New Connection

Connection Name:	MySQL running in AWS	Type a name for the connection
Connection Method:	Standard (TCP/IP)	Method to use to connect to the RDBMS

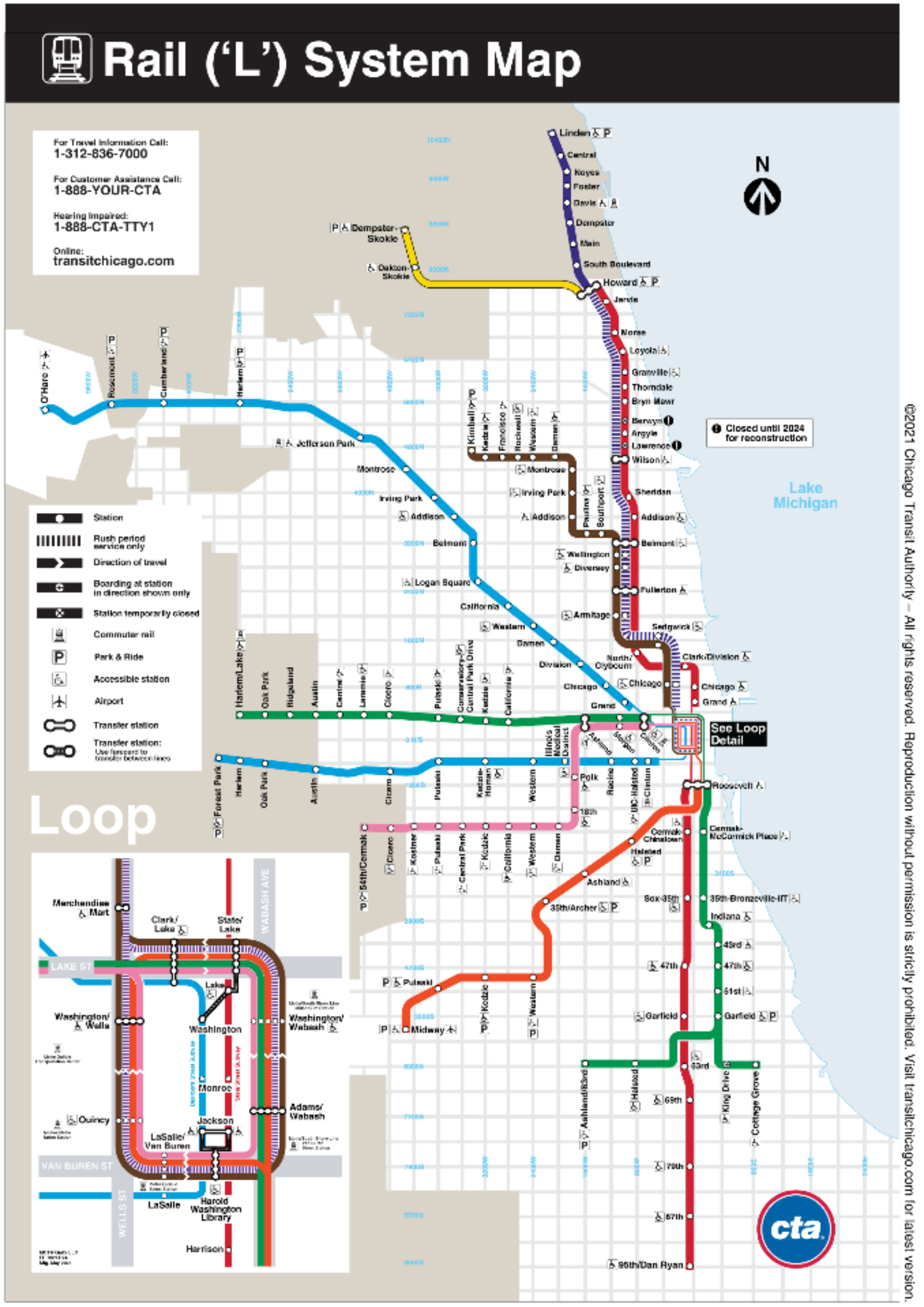
Parameters | SSL | Advanced

Hostname:	nu-cs-msa-mysql.cb1xaky37wq8.us-east-2.rds.amazonaws.com Port: 3306	Name or IP address of the server host - and TCP/IP port.
Username:	movielens-read-only	Name of the user to connect with.
Password:	Store in Vault ... Clear	The user's password. Will be requested later if it's not set.
Default Schema:		The schema to use as default schema. Leave blank to select it later.

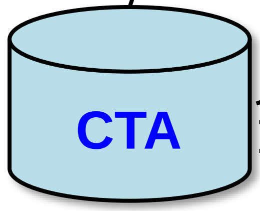


```
use movielens;

select count(*) from movies;

select count(*) from ratings;
```

That's it, thank you!


Rail ('L') System Map
For Travel Information Call: 1-312-836-7000
For Customer Assistance Call: 1-888-YOUR-CTA
Hearing Impaired: 1-888-CTA-TTY1
Online: transitchicago.com
Station
Rush period service only
Direction of travel
Boarding at station in direction shown only
Station temporarily closed
Commuter rail
Park & Ride
Accessible station
Airport
Transfer station
Transfer station: Use farecard to transfer between lines
Closed until 2024 for reconstruction
Lake Michigan
N
Loop
See Loop Detail
Linden
Howard
Dempster-Skokie
Oakton-Skokie
O'Hare
Jefferson Park
Kimball
Harlem/Lake
Forest Park
54th/Cermak
Midway
95th/Dan Ryan
Ashland/63rd
Cottage Grove
Merchandise Mart
Clark/Lake
State/Lake
Washington/Wells
Washington/Wabash
Quincy
LaSalle/Van Buren
Harold Washington Library
Adams/Wabash
Monroe
Jackson
LaSalle
Harrison
LAKE ST
VAN BUREN ST
WELLS ST
WABASH AVE
cta
©2021 Chicago Transit Authority – All rights reserved. Reproduction without permission is strictly prohibited. Visit transitchicago.com for latest version.

CTA database



Stations

Station_ID	Station_Name
40710	*Chicago/Franklin*
…	…

Stops

Stop_ID	Station_ID	Stop_Name	Direction	ADA	Latitude	Longitude
30137	*40710*	*Chicago (Kimball-Linden-bound)*	*N*	*1*	*41.89681*	*-87.635924*
30138	*40710*	*Chicago (Loop-bound)*	*S*	*1*	*41.89681*	*-87.635924*
…	…	…	…	…	…	…

Lines

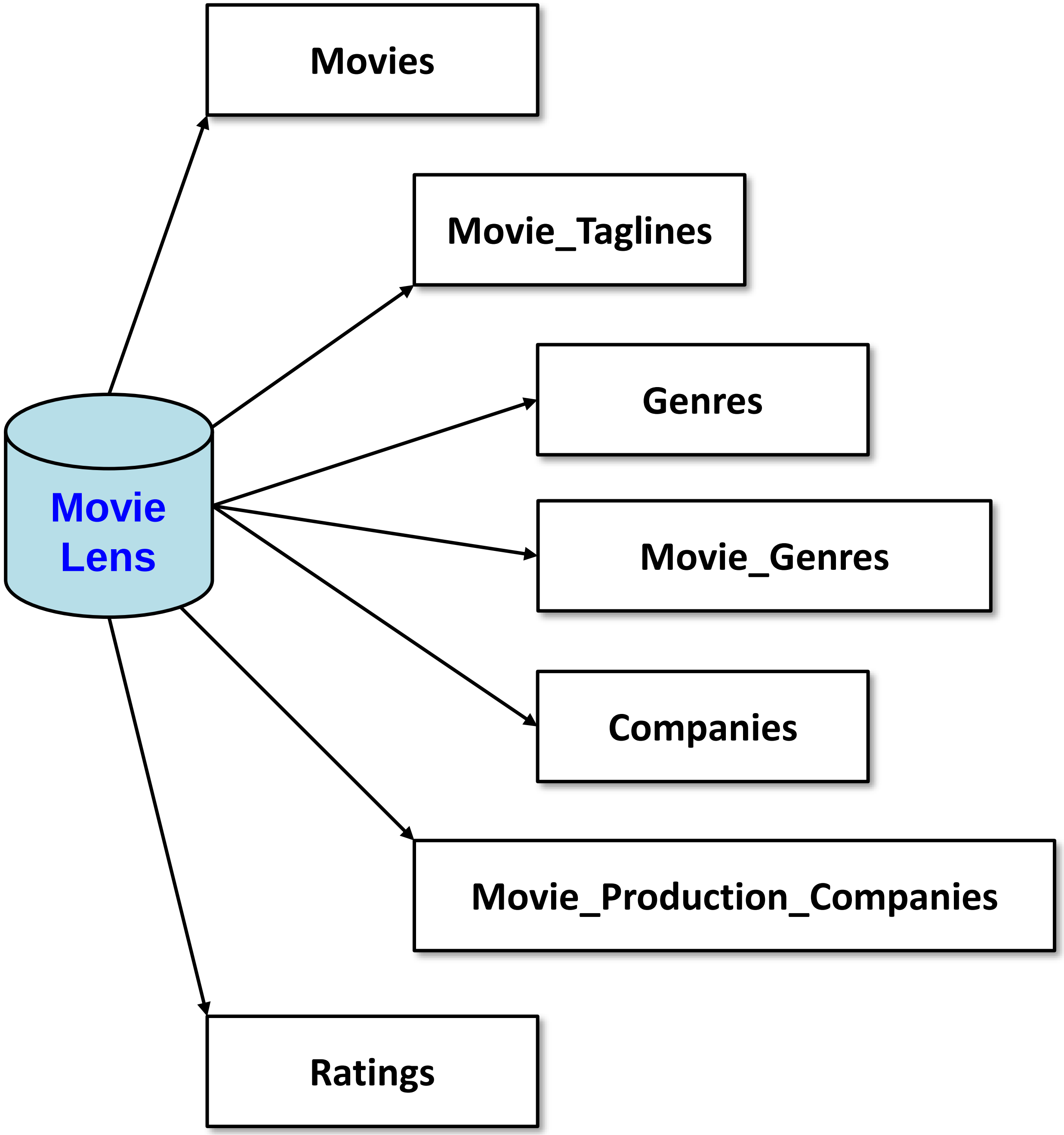
Line_ID	Color
…	…
4	Brown
5	Purple
6	Purple-Express
…	…

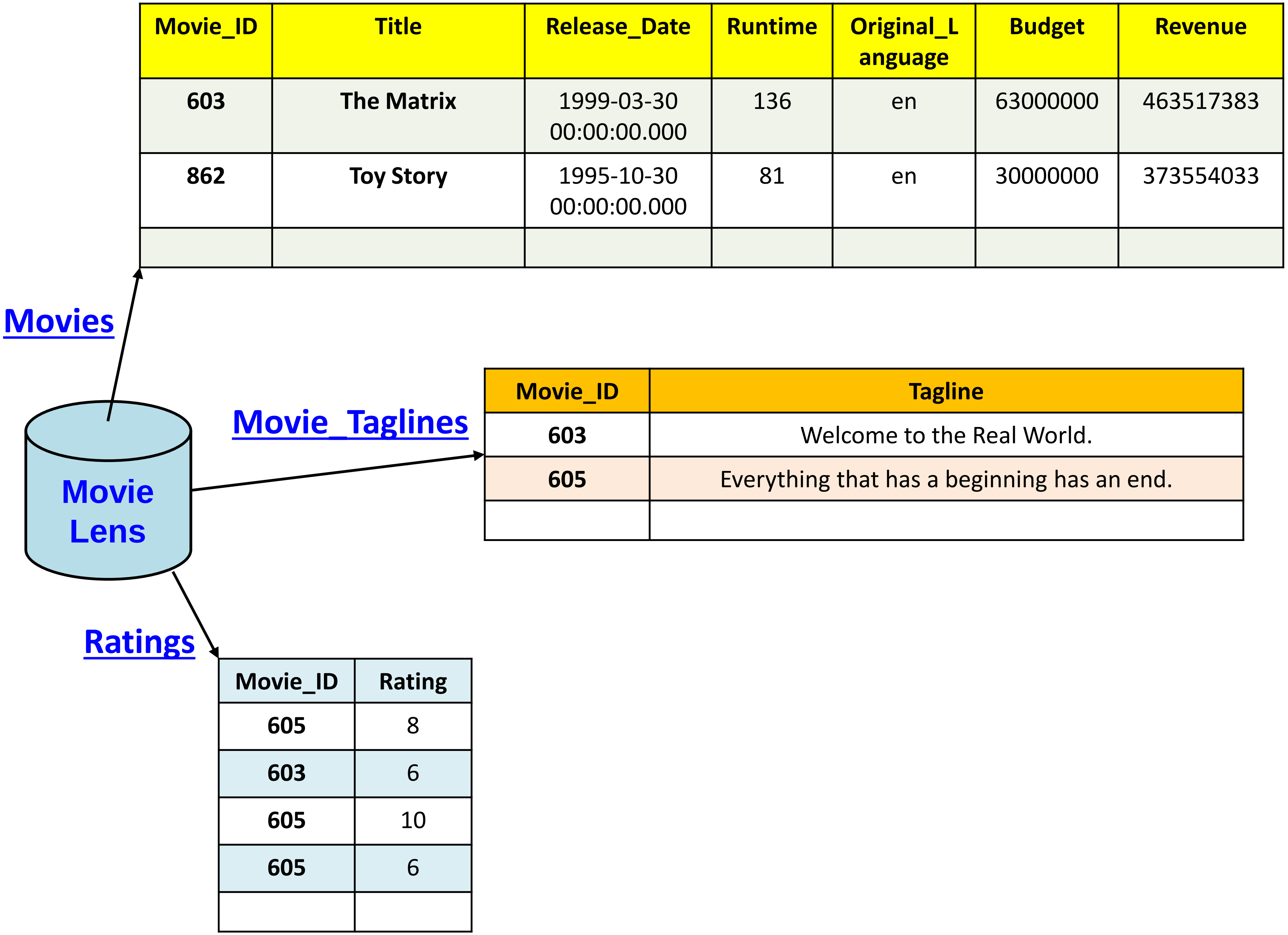
LinesPerStop

Stop_ID	Line_ID
…	…
30137	4
30137	6
30138	4
30138	6
…	…

Ridership

Station_ID	Ride_Date	Type_of_Day	Num_Riders
…	…	…	…
40710	2001-02-28 00:00:00.000	W	4206
…	…	…	…


Movie
Lens
Movies
Movie_Taglines
Genres
Movie_Genres
Companies
Movie_Production_Companies
Ratings



Movies

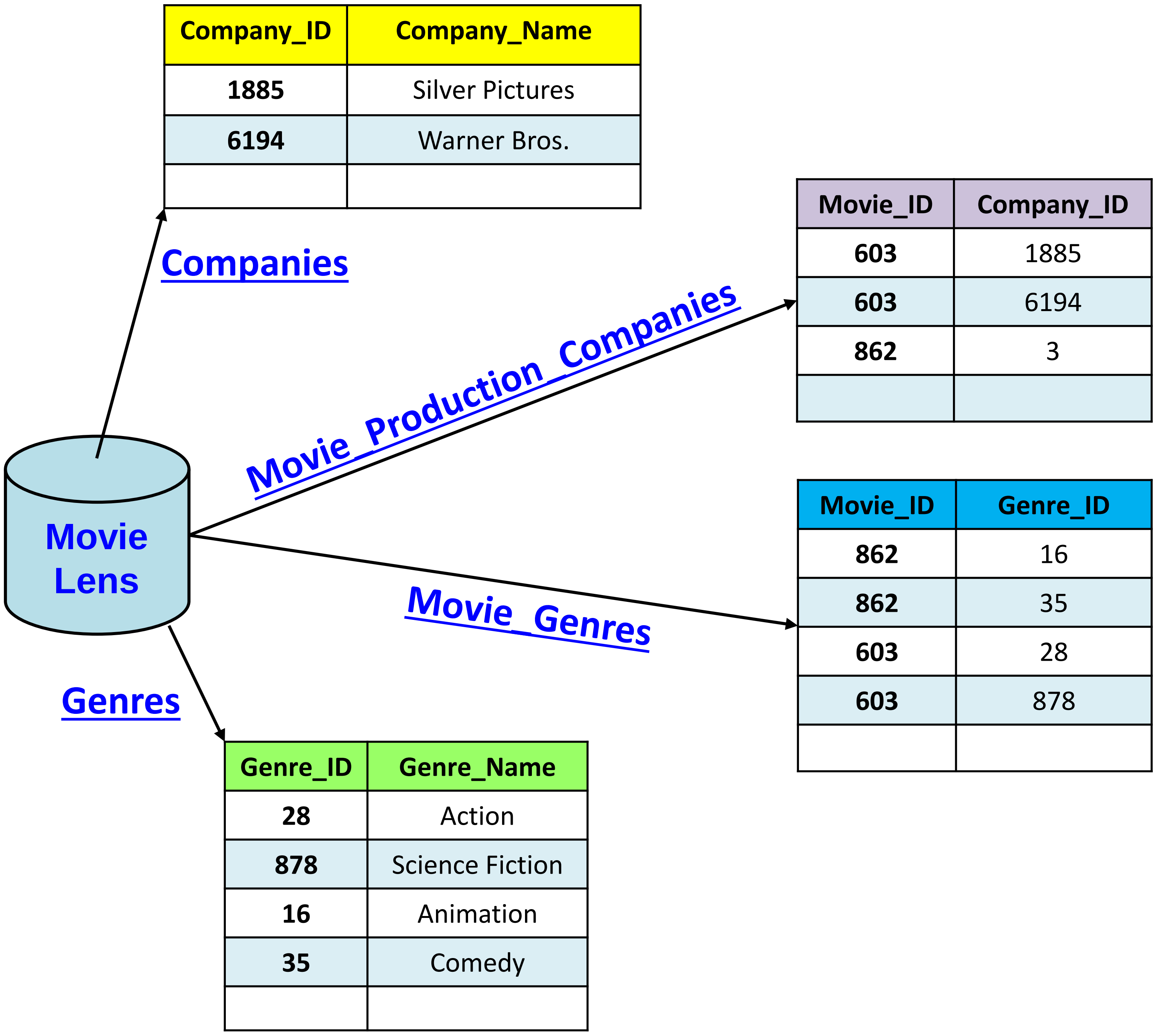
Movie_ID	Title	Release_Date	Runtime	Original_Language	Budget	Revenue
603	**The Matrix**	1999-03-30 00:00:00.000	136	en	63000000	463517383
862	**Toy Story**	1995-10-30 00:00:00.000	81	en	30000000	373554033

Movie_Taglines

Movie_ID	Tagline
603	Welcome to the Real World.
605	Everything that has a beginning has an end.

Ratings

Movie_ID	Rating
605	8
603	6
605	10
605	6



Company_ID	Company_Name
1885	Silver Pictures
6194	Warner Bros.

Movie_ID	Company_ID
603	1885
603	6194
862	3

Movie_ID	Genre_ID
862	16
862	35
603	28
603	878

Genre_ID	Genre_Name
28	Action
878	Science Fiction
16	Animation
35	Comedy