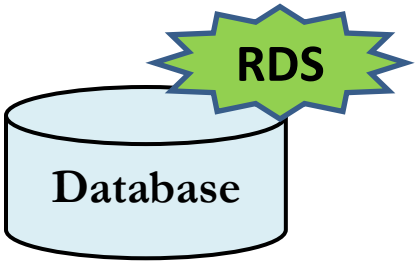
Intro to Relational Databases and SQL

- Why relational databases?
- Is SQL still relevant?
- Retrieving data with Select queries

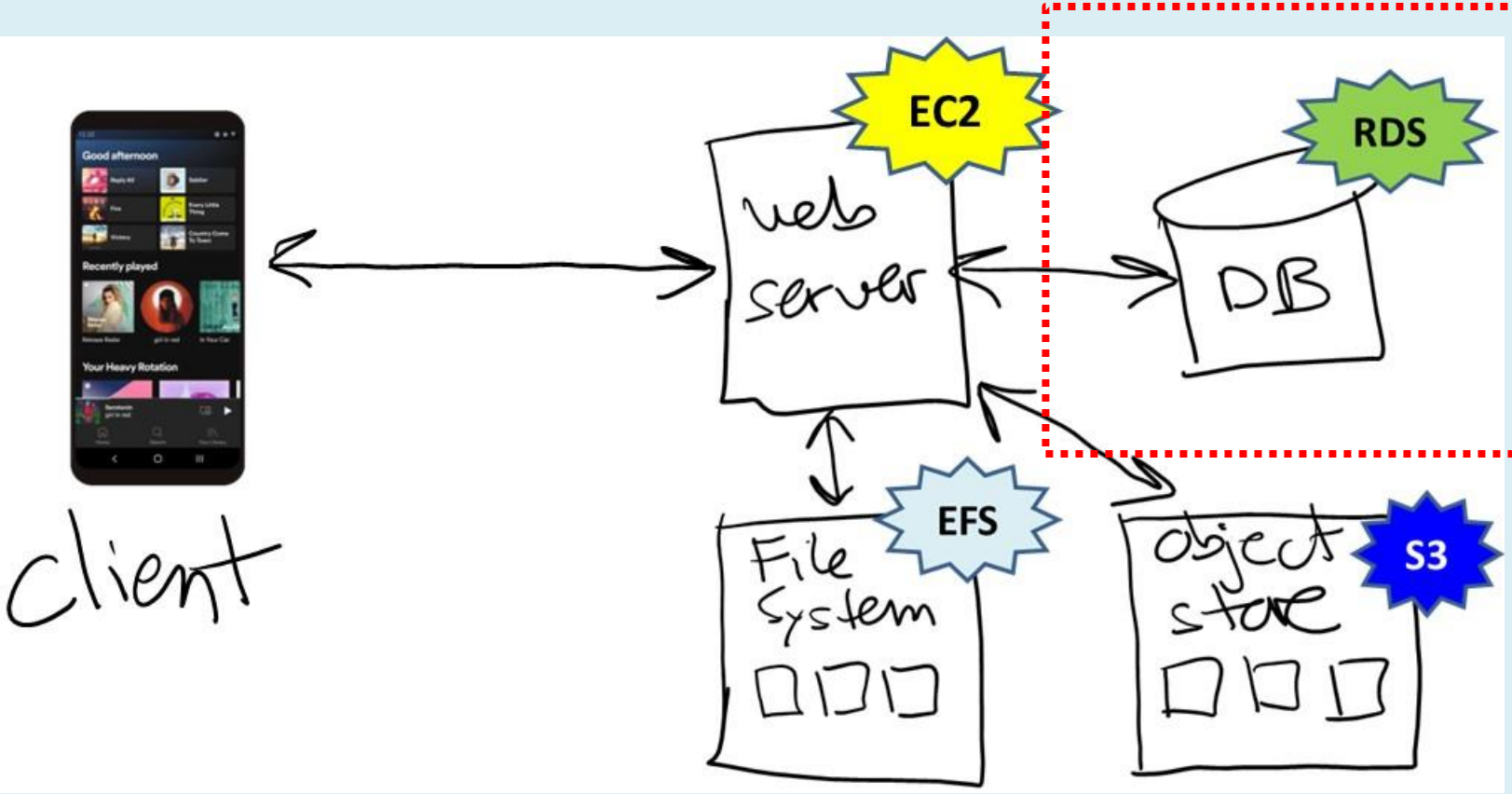
Critical infrastructure



- **Databases are critical infrastructure**

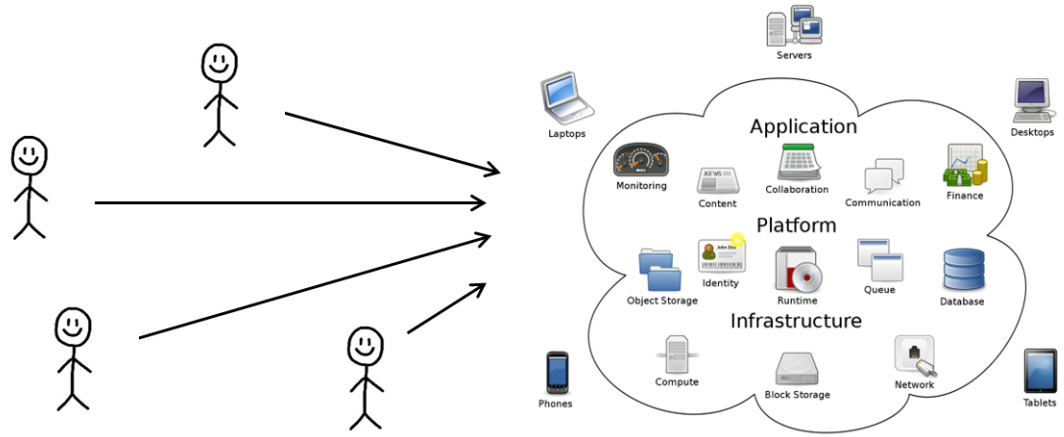
- **Most companies would go out of business if their database is lost / corrupted**

Relational Databases



Relational databases --- pros and cons

- **Advantages to using relational databases?**
  - *Data is structured, clean (error-free), and well-organized*
  - *SQL is a powerful, high-level programming language*
  - *Database management systems (DBMS) are fast, efficient, and reliable*



Disadvantages?

- **Disadvantages of relational databases?**

This is an advantage as well as disadvantage, as we can program against it if it is fixed but at the same time, not flexible to change anything.

  - *Database design (schema) is rigid --- hard to evolve/change*

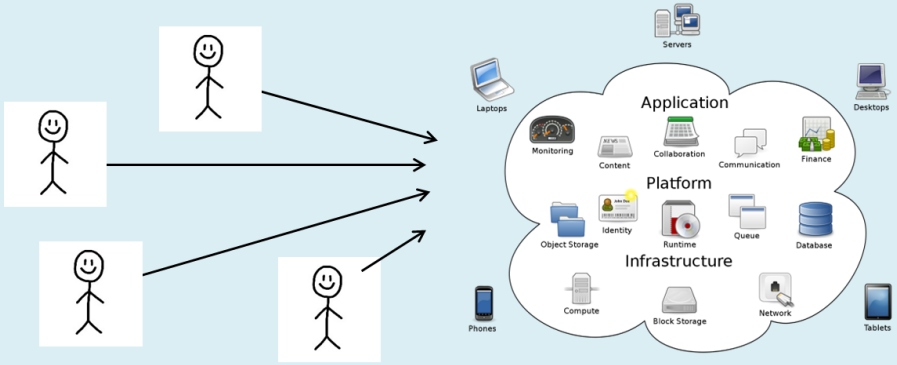
Need to clean and structure data before importing into database and cleaning and structuring real data is very hard and messy

  - *Data cleaning for import is HARD*

This means, if change something in one replica, it needs to appear across all the replicas. This becomes a bottleneck to do for such a reliable system and requires lots of engineering effort.

  - *Difficult to scale to large amounts of data / number of users*

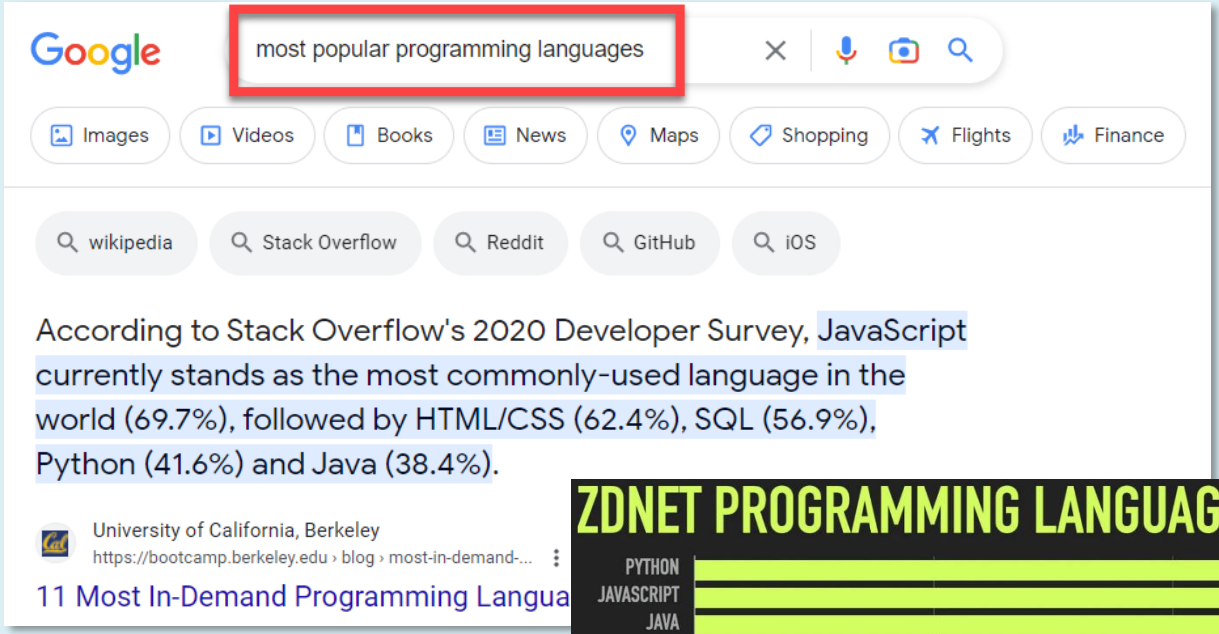
*Databases rely on a **centralized model**, and so replication is more difficult to engineer*

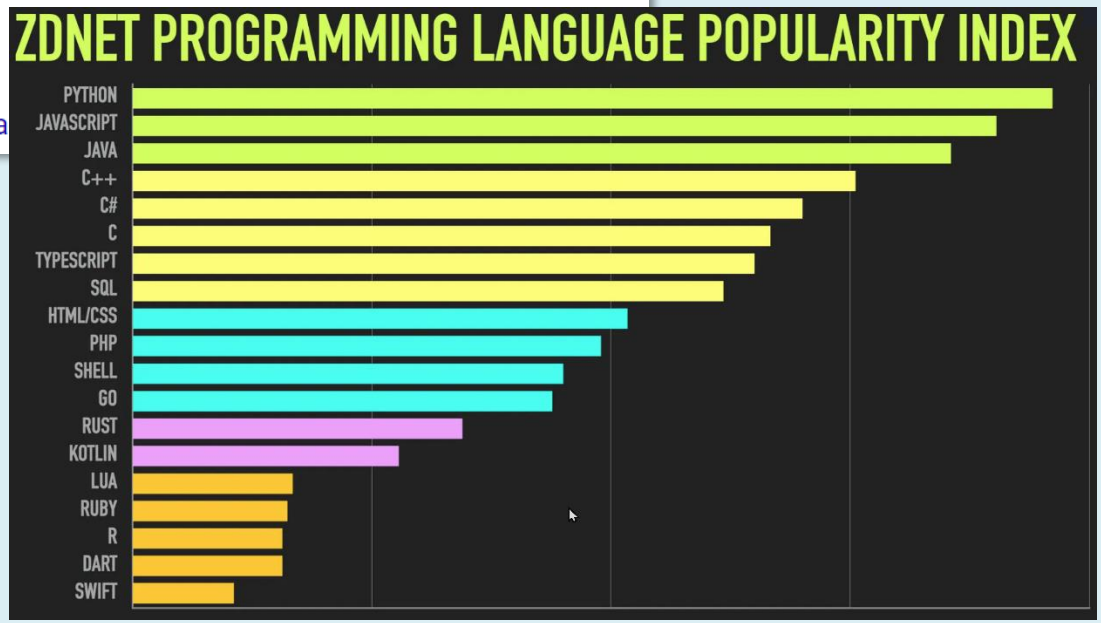


SQL

- **Structured Query Language**
- **The language of relational databases**
- **One of the most important programming languages you need to know**

Is SQL still relevant?





Declarative Programming

***Def**: **declarative programming** is a paradigm where you express **what** you want, but **not how** — you let the system decide how best to do things.*

- ***THE*** **best example of declarative programming?**

Example

```
SELECT Title, PubDate FROM Books
  WHERE Author = 'Stephen King'
  ORDER BY PubDate DESC, Title ASC;
```

Title	PubDate
"Doctor Sleep"	2014
"Revival: A Novel"	2014
"Under the Dome: Part 2"	2014
"Joyland"	2013
"Salem's Lot"	2013
"Under the Dome"	2013
…	…

But here since SQL is declarative, it handles all these details by itself.

*Normally we would have to handle the details of the **file format**… **data structure**… **searching algorithm**… **sorting algorithm**…*

- **The SQL language is big, with 2 major sub-languages:**
  - ***DDL***: *data definition language*

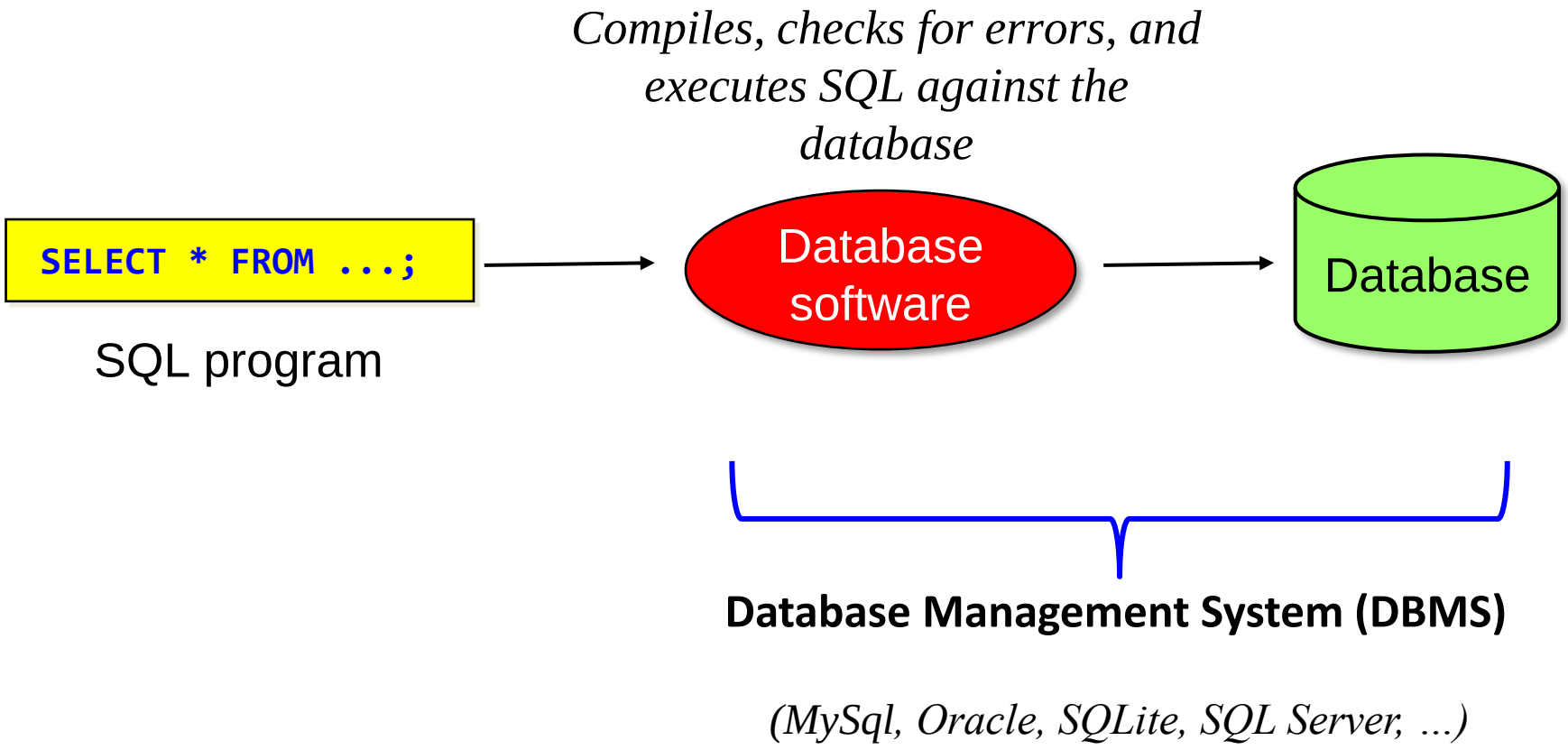
For creating a database
- *Create Table*
- *Create Index*
- *Create Stored Procedure*
- *…*

  - ***DML***: *data manipulation language*

For manipulating data in database:
- *Select * From … Where …*
- *Insert*
- *Update*
- *Delete*
- *…*

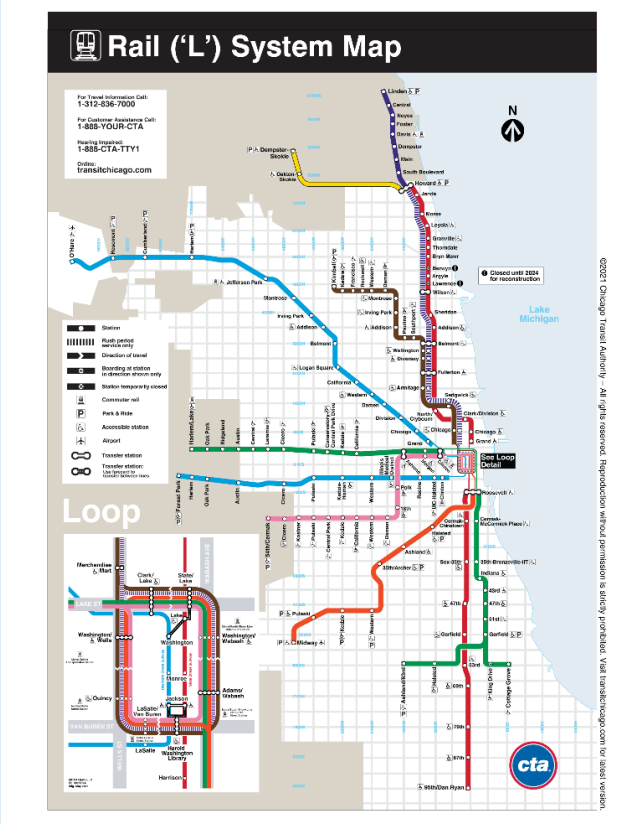
SQL + DBMS

- **SQL programs are compiled and executed like other languages**



Database example: CTA

- We have some CTA ridership data (L stations) that we need to analyze…



stations.csv - Notepad2

File Edit View Settings ?

```
1 40010,Austin-Forest Park
2 40020,Harlem-Lake
3 40030,Pulaski-Lake
4 40040,Quincy/Wells
5 40050,Davis
6 40060,Belmont-O'Hare
7 40070,Jackson/Dearborn
8 40080,Sheridan
9 40090,Damen-Brown
10 40100,Morse
11 40120,35th/Archer
12 40130,51st
13 40140,Dempster-Skok
14 40150,Pulaski-Cerma
15 40160,LaSalle/Van B
```

ridership.csv - Notepad2

File Edit View Settings ?

```
1 41280,2017-12-22 00:00:00.000,W,6104
2 41000,2017-12-18 00:00:00.000,W,3636
3 40280,2017-12-02 00:00:00.000,A,1270
4 40140,2017-12-19 00:00:00.000,W,1759
5 40690,2017-12-03 00:00:00.000,U,499
6 41660,2017-12-30 00:00:00.000,A,8615
7 40180,2017-12-17 00:00:00.000,U,442
8 40250,2017-12-02 00:00:00.000,A,1353
9 40120,2017-12-07 00:00:00.000,W,3353
10 41420,2017-12-19 00:00:00.000,W,6034
11 40270,2017-12-16 00:00:00.000,A,887
12 41450,2017-12-27 00:00:00.000,W,9639
13 41210,2017-12-07 00:00:00.000,W,3210
14 40010,2017-12-03 00:00:00.000,U,641
15 41160,2017-12-31 00:00:00.000,U,621
16 40720,2017-12-26 00:00:00.000,W,613
17 40330,2017-12-21 00:00:00.000,W,10683
18 40540,2017-12-22 00:00:00.000,W,4861
```

CTA database (subset)

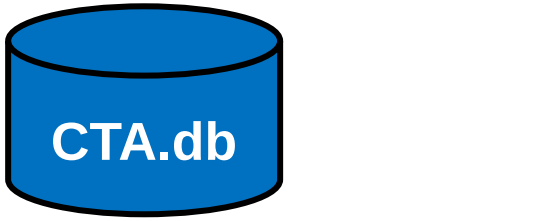


table: Stations

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

table: Ridership

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
…	…	…	…

Example #1

Stations

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

- **The closest L station is "Noyes"**
- **What is the station ID?**
  - *Answer: 40400*

```
select Station_ID from Stations where Station_Name = 'Noyes';
```

To Run with Docker

sqlite3 < main.sql

< == read from something

- **Observations…**
  - *SQL language is case insensitive (select or Select or SELECT)*
  - *Data is case sensitive (try 'noyes')*

Select

- For retrieving data from a database
- General format:

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

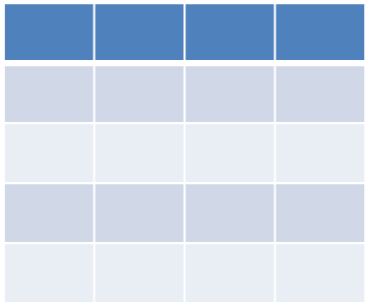
optional

Search Criteria

Search Criteria when the table is grouped

Sorting

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



Example #2

- **List all the L stations in alphabetical order:**

```
select Station_ID, Station_Name
from   Stations
order by Station_Name Asc;
```

Stations

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

Example #3

- **What is avg daily ridership @ Noyes (station 40400)?**
  - *Answer: 592.793*

```
select avg( ??? ) from Ridership where Station_ID = ???;
```

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

- **Notes:**
  - *SQL is a programming language with built-in functions*
  - *Use **round(value, places)** if you want to round the answer*

Query

```
select round(avg(Num_Riders), 3)
from   Ridership
where  Station_ID = 40400;   -- 'Noyes' station
```

Computation

- **Like most languages, SQL can perform computation**

- **Example:**
  - *What do you think this computes?*

```
Select
  cast((Select sum(Num_Riders) From Ridership Where Station_ID = 40400) as float) /
  cast((Select sum(Num_Riders) From Ridership) as float) *
  100.0;
```

SQL is a full programming language

- **SQL has variables, if stmts, loops, functions, classes, etc.**

```
Declare @total as float;
Declare @noyes as float;

Set @total = (Select sum(Num_Riders) From Ridership);
Set @noyes = (Select sum(Num_Riders) From Ridership Where Station_ID = 40400);

Select @noyes/@total * 100.0;
```

```
If @noyes is NULL  -- incorrect station id
  Select 0
Else
  Select @noyes/@total * 100.0;
```

That's it, thank you!