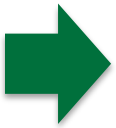
CS 61:
Database Systems

Introduction to the relational model

Adapted from Silberschatz, Korth, and Sundarshan unless otherwise noted

Agenda



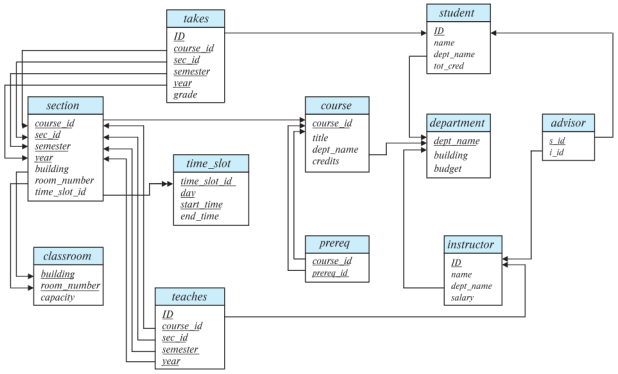
1. Big picture of relational database design
2. Relational algebra
3. Intro to SQL SELECT statement
4. NYC Open Data

Big picture of relational database design

Relational Database Management System

- Normally represented graphically as a cylinder
- Holds data in relations (tables)

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000



Relations

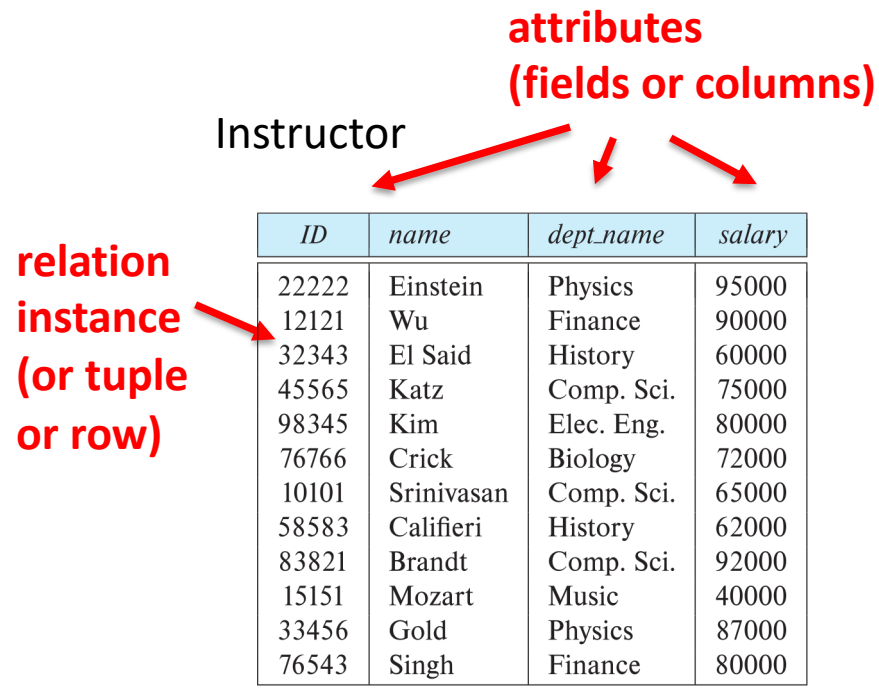
- Each relation holds data about people, places, things or events (nouns)
- Tables consist of rows and columns
- Each row (tuple) represents one person, place, thing, or event
- Each column represents one attribute about a person, place, thing, or event (e.g. name)
- A column (FK) can refer to a column (PK) in another table, creating a relationship between tables

Database schema

- Logical collection of tables and relationships
- Minimizes storing multiple copies of data
- Look up additional data in another table if needed using key

Relational databases store data in relations (tables) made up of attributes

Instructor relation (table)



ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

Data in a relational database

- Databases store data in **relations** (tables)
- Relations are made up of **relation instances** (rows)
- Relation instances comprised of **attributes** (columns)
- Relation and attribute names are unique

Attributes

- The set of allowed values for each attribute is called the **domain** of the attribute
- Attribute values are (normally) required to be **atomic**; that is, indivisible
- Order of attributes is irrelevant (a Set)
- The special value **NULL** is a member of every domain. Indicates that the value is “unknown”
- We will see soon that NULL causes complications in some operations

Relation instances (rows or tuples)

- Each relation instance represents one person, place, thing, or event
- Order of instances is irrelevant
- Each instance must be uniquely identified (no duplicate rows, at least in theory)

Relations in a relational database must conform to eight rules

Table characteristics

6 rows (tuples) with 3 columns (attributes) for each row

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

Each row describes one department

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

Each column represents a different attribute of a department (e.g., ID, Name, Building) and each column has a different name

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set
3. Each column represents an attribute, and each column has distinct name

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

Single entry in each cell

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set
3. Each column represents an attribute, and each column has distinct name
4. Each intersection of a row and column represents a single data value

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

In column 1 all entries are numeric, in other columns each entry is character data

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set
3. Each column represents an attribute, and each column has distinct name
4. Each intersection of a row and column represents a single data value
5. All values in a column must conform to the same data format

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

Domain is positive integers for column 1, alphanumeric characters for others

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set
3. Each column represents an attribute, and each column has distinct name
4. Each intersection of a row and column represents a single data value
5. All values in a column must conform to the same data format
6. Each column has a specific range of values known as the **attribute domain**

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

Departments not ordered in any particular fashion, except CS is first ;-)

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set
3. Each column represents an attribute, and each column has distinct name
4. Each intersection of a row and column represents a single data value
5. All values in a column must conform to the same data format
6. Each column has a specific range of values known as the **attribute domain**
7. The order of the rows and columns is immaterial to the DBMS

Adapted from Coronel and Morris

Relations in a relational database must conform to eight rules

Table characteristics

DepartmentID is a *Primary Key* (PK), it can uniquely identify each row

No two rows can be *exactly* the same

Department table

DepartmentID	DepartmentName	DepartmentBuilding
1	Computer Science	Sudikoff Lab
2	Biology	Life Sciences Center
3	English	Sanborn
4	Chemistry	Burke
5	Government	Silsby
6	Engineering	Thayer

1. Each table is perceived as a two-dimensional structure of rows and columns
2. Each row (tuple) represents a single entity occurrence within the entity set
3. Each column represents an attribute, and each column has distinct name
4. Each intersection of a row and column represents a single data value
5. All values in a column must conform to the same data format
6. Each column has a specific range of values known as the **attribute domain**
7. The order of the rows and columns is immaterial to the DBMS
8. Each table must have an attribute or combination of attributes that uniquely identifies each row

NOTE: a value of NULL means the value is not known or empty; Primary keys cannot be null

Adapted from Coronel and Morris

Highlander theory of database design: "There can be only one! (copy of the data)"

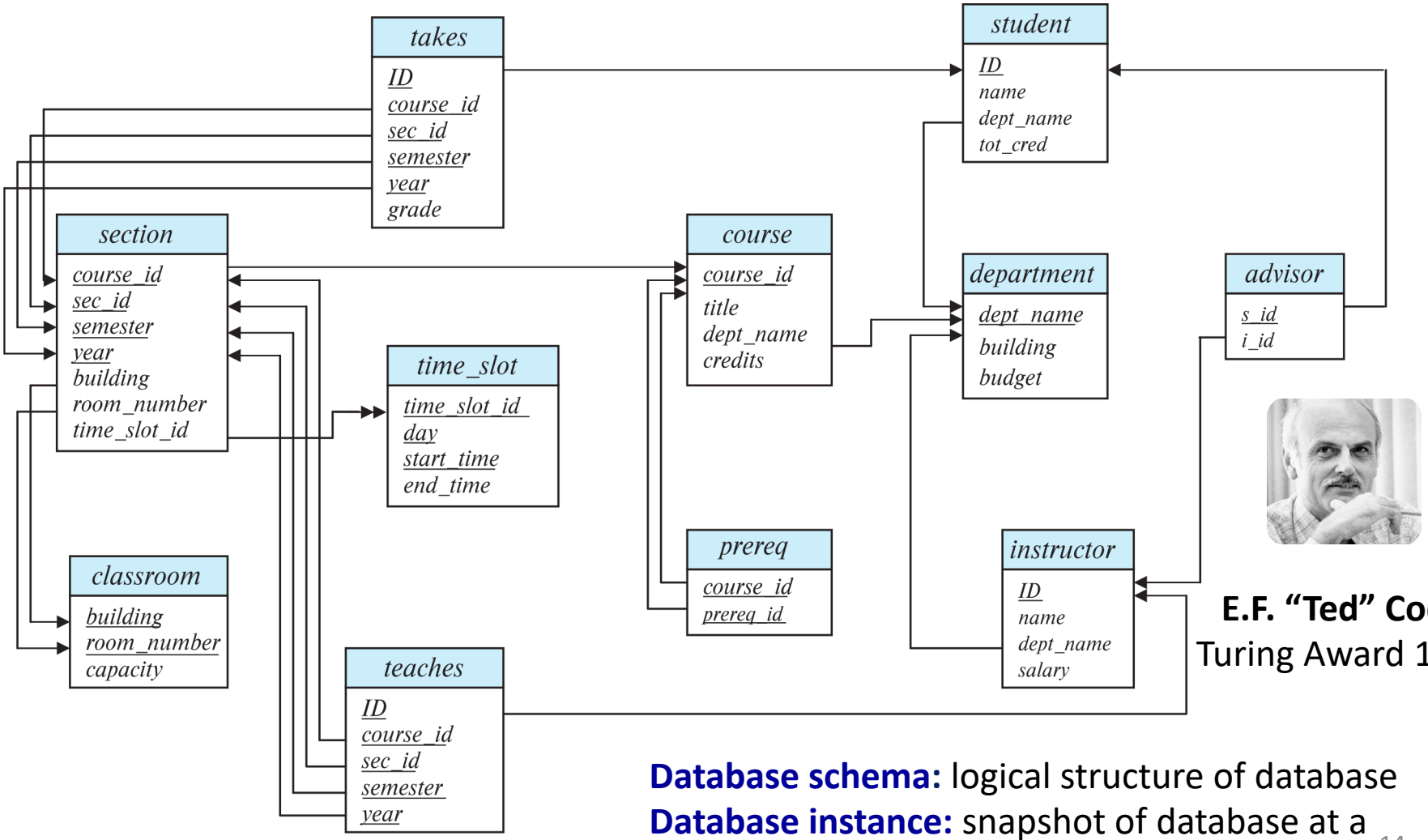
Avoid storing the same data multiple times, store it once!

- We will discuss this idea further when we cover normalization
- For now tables hold data about one type of entity (e.g., customer), each row in the table is an instance of that thing (e.g., Sally Jones)

- Each table holds data about a type of entity: a person, place, thing or event
- Avoid storing the same data in multiple tables!
- Example:
  - Do not store a customer's address in multiple tables
  - Instead create one table that represents customers and store their address as columns in that single table
  - Other tables that need the customer's address look it up in this table
  - If address changes, only one update needed

Look up data in other tables when needed

Database schema diagram



E.F. "Ted" Codd
Turing Award 1981

Database schema: logical structure of database
Database instance: snapshot of database at a point in time

Agenda

1. Big picture of relational database design
2. Relational algebra
3. Intro to SQL SELECT statement
4. NYC Open Data

Relational algebra allows us to work with data in relations (tables)

Mathematically

- Let A_1, A_2, …, A_n be a set of n **attributes**
- Let $R = (A_1, A_2, \ldots, A_n)$ be the set of attributes in the schema of relation ***r***

  Example:

  instructor = (*ID, name, dept_name, salary*)
- A relation instance *r* defined over schema *R* is denoted by *r* (*R*)

Implementation

- The current values (**relation instances**) of a relation are specified by a table
- An element ***t*** of relation ***r*** is called a **tuple** and is represented by a table *row*
- Duplicates tuples (rows) are not allowed in a relation (but are in table!)

  If t_1 and t_2 are tuples in r, then $t_1 \neq t_2$

Project: returns a subset of attributes from relation *r*

Project notation: $\Pi_{A_1, A_2, A_3 \ldots A_k}(r)$

project

$\Pi_{ID,\ name,\ salary}(instructor)$

relation *r*

attributes

What *columns* of a relation do we want

instructor relation

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

result

dept_name left out

ID	*name*	*salary*
10101	Srinivasan	65000
12121	Wu	90000
15151	Mozart	40000
22222	Einstein	95000
32343	El Said	60000
33456	Gold	87000
45565	Katz	75000
58583	Califieri	62000
76543	Singh	80000
76766	Crick	72000
83821	Brandt	92000
98345	Kim	80000

Select: returns tuples from relation *r* that satisfy predicate *p*

Select notation: $\sigma_p(r)$

select

$\sigma_{dept\_name=\text{“Physics”}}(instructor)$

relation *r*

predicate *p*

What *rows* of relation do we want

instructor relation

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

result

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
33456	Gold	Physics	87000

- In selection predicate can use:
  =, ≠, >, ≥, <, ≤
- Can combine several predicates:
  ^ (and), v (or), ¬ (not)

Example:

$\sigma_{dept\_name=\text{“Physics”} \wedge salary > 90{,}000}(instructor)$

The result of an operation is a relation, so we can combine them into an expression

Relational algebra expression

project

Select returns relation

relation *r*

$$\Pi_{name}(\sigma_{dept\_name = \text{“Physics”}}(instructor))$$

attribute

predicate *p*

instructor relation

ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

result

name
Einstein
Gold

“Find the name of instructors in the Physics department”

- Select eliminated rows we do not want
- Project eliminated columns we do not want
- Operation returns a relation (here with 2 tuples)

Agenda

1. Big picture of relational database design
2. Relational algebra
3. Intro to SQL SELECT statement
4. NYC Open Data

SQL Select command has three parts, like relational algebra expression

- SQL Select returns a relation with specified attributes from one or more relations with tuples matching provided criteria
- A typical SQL Select query has the form:

SELECT $A_1, A_2, \ldots, A_n$
FROM $r_1, r_2, \ldots, r_m$
WHERE P

Specify the attributes (columns) we want (like Project)

From the relations (tables) we want, today only one

Return tuples meeting some requirement (like Select)

SQL Select command has three parts, like relational algebra expression

Generic SELECT

SELECT $A_1, A_2, \ldots, A_n$
FROM $r_1, r_2, \ldots, r_m$
WHERE P

Example SELECT

SELECT name
FROM instructor
WHERE dept_name = 'Physics';

Resulting relation

name
Einstein
Gold

Equivalent relational algebra

$$\Pi_{name}(\sigma_{dept\_name = \text{“}physics\text{”}}(instructor))$$

SQL Select command

- Three parts:
  1. **SELECT** attributes – what columns we want (instructor name)
  2. **FROM** – relation to use (instructor relation)
  3. **WHERE** – criteria for selecting tuples (dept_name = ‘Physics’)
- SQL command capitalization does not matter (Select == select == SELECT)
- Convention is to capitalize SQL commands, but not required
- Relation/attribute names in MySQL not case sensitive but are in some databases
- Use a single quote for strings in SQL (this may bite you, if you get an error: Unknown column “Physics”, it did! Make it 'Physics' (single quote) instead)

Agenda

1. Big picture of relational database design
2. Relational algebra
3. Intro to SQL SELECT statement
4. NYC Open Data

NYC is made up of five boroughs

Five NYC boroughs

Manhattan

The Bronx

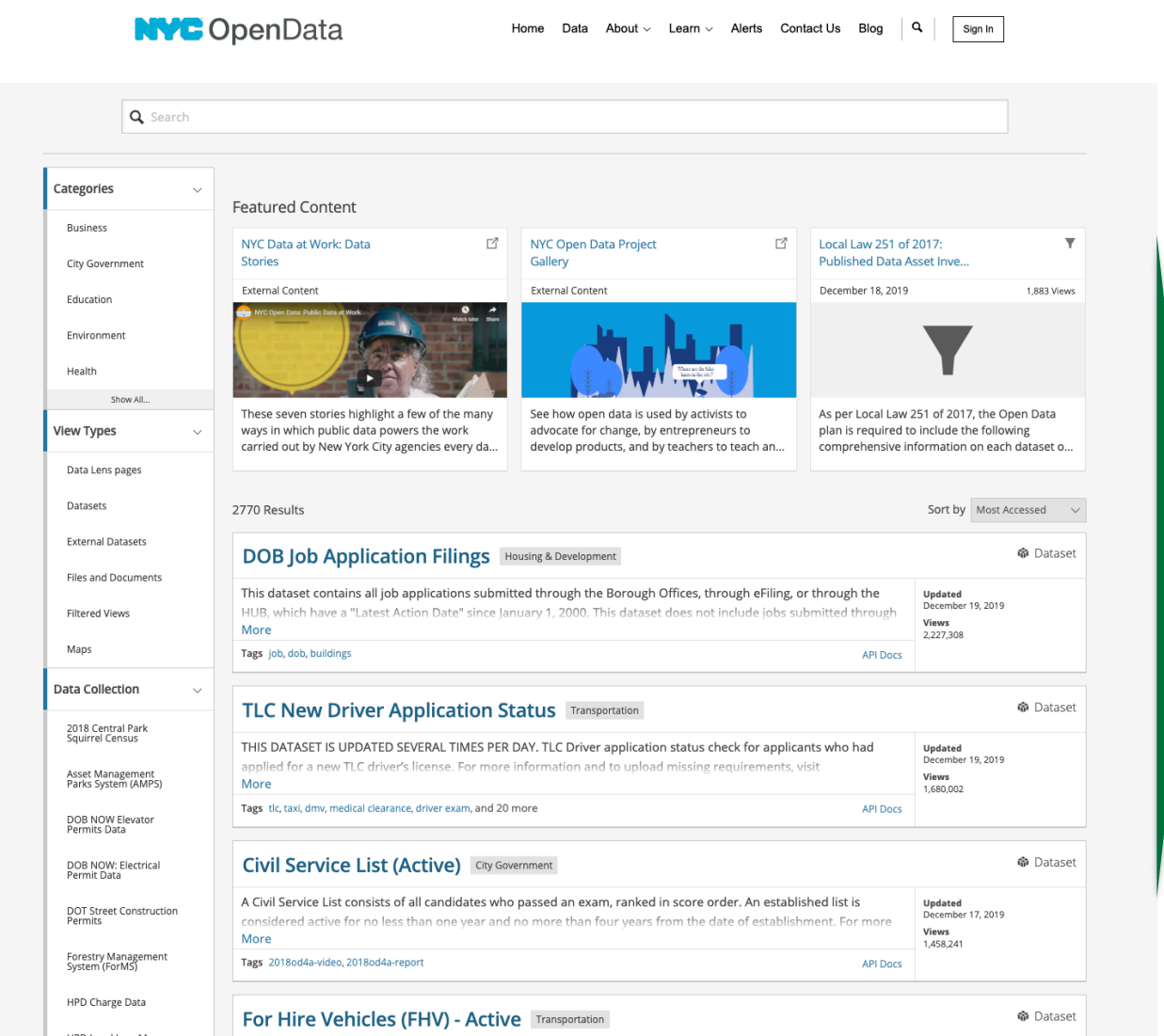
Queens

Brooklyn

Staten Island

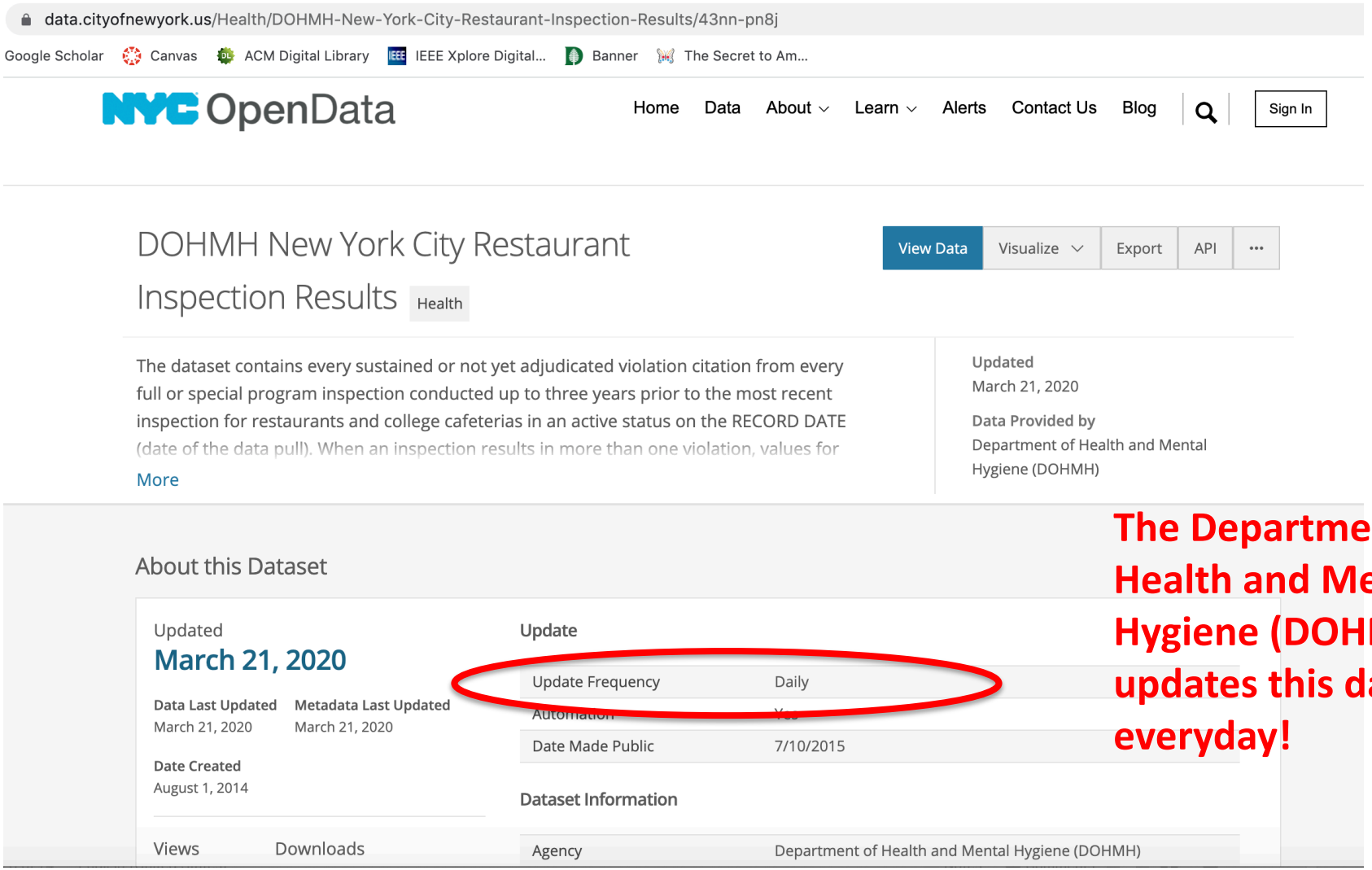
New York makes an incredible amount of data publicly available in NYC Open Data

https://data.cityofnewyork.us/browse?sortBy=most_accessed



Lots of data collected by NYC (and other cities) is freely available

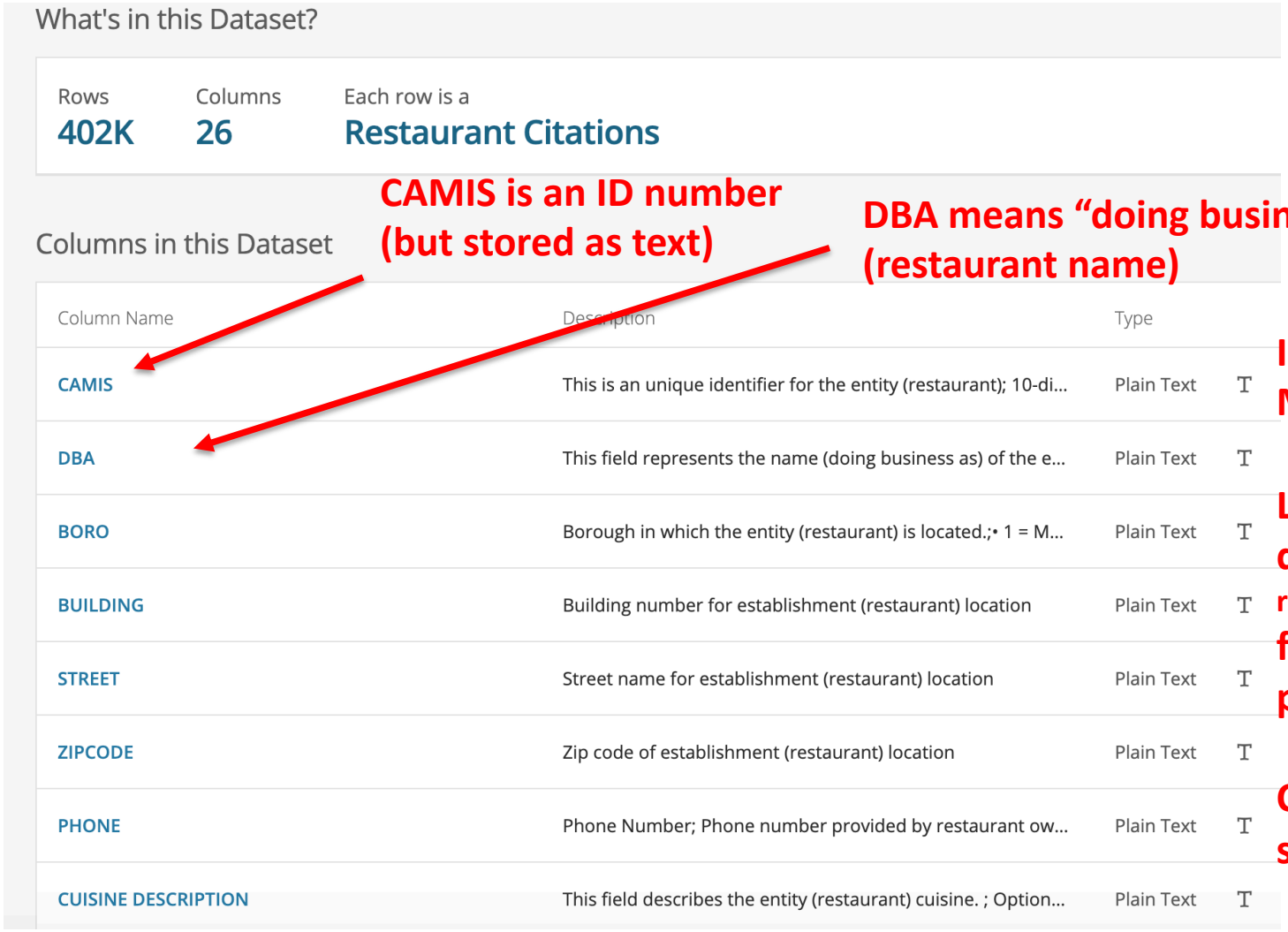
One data set NYC publishes contains all restaurant health inspections



The Department of Health and Mental Hygiene (DOHMH) updates this data set everyday!

Source: https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j

NYC provides a "data dictionary" that describes each column



What's in this Dataset?

Rows	Columns	Each row is a
402K	26	Restaurant Citations

CAMIS is an ID number (but stored as text)

DBA means "doing business as" (restaurant name)

Columns in this Dataset

Column Name	Description	Type
CAMIS	This is an unique identifier for the entity (restaurant); 10-di...	Plain Text
DBA	This field represents the name (doing business as) of the e...	Plain Text
BORO	Borough in which the entity (restaurant) is located.;• 1 = M...	Plain Text
BUILDING	Building number for establishment (restaurant) location	Plain Text
STREET	Street name for establishment (restaurant) location	Plain Text
ZIPCODE	Zip code of establishment (restaurant) location	Plain Text
PHONE	Phone Number; Phone number provided by restaurant ow...	Plain Text
CUISINE DESCRIPTION	This field describes the entity (restaurant) cuisine. ; Option...	Plain Text

I loaded this data into MySQL

Load it into your local database with restaurant_inspections.sql from the course web page

Can also find it on sunapee

Source: https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j

First get a feel for the data by selecting all attributes

SELECT command (query)

```
USE nyc_data;

SELECT * FROM restaurant_inspections LIMIT 100;
```

Tell MySQL which schema (database) to use

- **Commands end with ;**
- **Can run multiple commands, like a program**

*** means return all attributes (columns)**

Table "restaurant_inspections" has results of 397,854 health inspections

- **Each restaurant may have been inspected multiple times over the years**
- **Only active restaurants listed in this dataset**

No WHERE clause so all tuples (rows) match select criteria

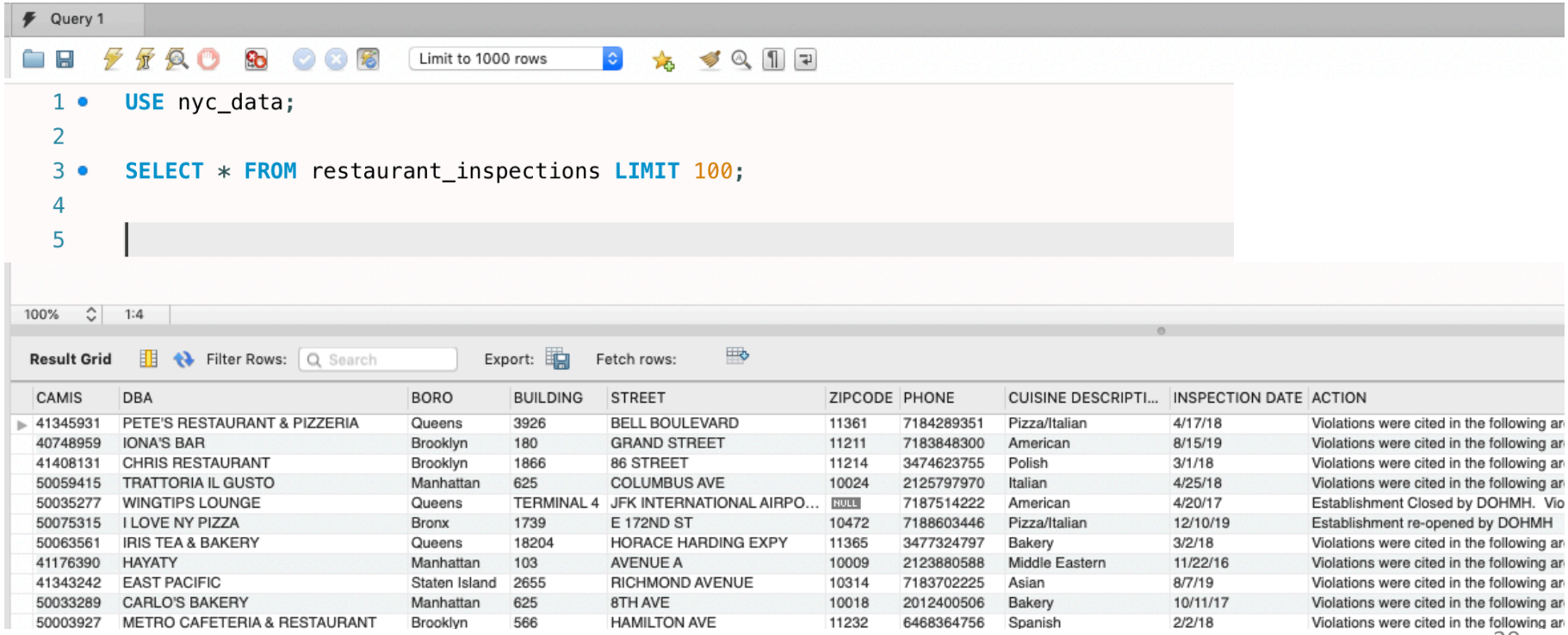
Only return the first 100 rows

First get a feel for the data by selecting all attributes

SELECT command (query)

```
USE nyc_data;

SELECT * FROM restaurant_inspections LIMIT 100;
```



CAMIS	DBA	BORO	BUILDING	STREET	ZIPCODE	PHONE	CUISINE DESCRIPTI...	INSPECTION DATE	ACTION
41345931	PETE'S RESTAURANT & PIZZERIA	Queens	3926	BELL BOULEVARD	11361	7184289351	Pizza/Italian	4/17/18	Violations were cited in the following ar
40748959	IONA'S BAR	Brooklyn	180	GRAND STREET	11211	7183848300	American	8/15/19	Violations were cited in the following ar
41408131	CHRIS RESTAURANT	Brooklyn	1866	86 STREET	11214	3474623755	Polish	3/1/18	Violations were cited in the following ar
50059415	TRATTORIA IL GUSTO	Manhattan	625	COLUMBUS AVE	10024	2125797970	Italian	4/25/18	Violations were cited in the following ar
50035277	WINGTIPS LOUNGE	Queens	TERMINAL 4	JFK INTERNATIONAL AIRPO...	NULL	7187514222	American	4/20/17	Establishment Closed by DOHMH. Vio
50075315	I LOVE NY PIZZA	Bronx	1739	E 172ND ST	10472	7188603446	Pizza/Italian	12/10/19	Establishment re-opened by DOHMH
50063561	IRIS TEA & BAKERY	Queens	18204	HORACE HARDING EXPY	11365	3477324797	Bakery	3/2/18	Violations were cited in the following ar
41176390	HAYATY	Manhattan	103	AVENUE A	10009	2123880588	Middle Eastern	11/22/16	Violations were cited in the following ar
41343242	EAST PACIFIC	Staten Island	2655	RICHMOND AVENUE	10314	7183702225	Asian	8/7/19	Violations were cited in the following ar
50033289	CARLO'S BAKERY	Manhattan	625	8TH AVE	10018	2012400506	Bakery	10/11/17	Violations were cited in the following ar
50003927	METRO CAFETERIA & RESTAURANT	Brooklyn	566	HAMILTON AVE	11232	6468364756	Spanish	2/2/18	Violations were cited in the following ar

Source: https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j

Select can specify the rows we want and sort them

SELECT command (query)

Query 1

Limit to 1000 rows

```
1  use nyc_data;
2
3  select * from restaurant_inspections
4  where DBA is not null
5  order by DBA asc
6  limit 100;
7
8
```

100% 1:6

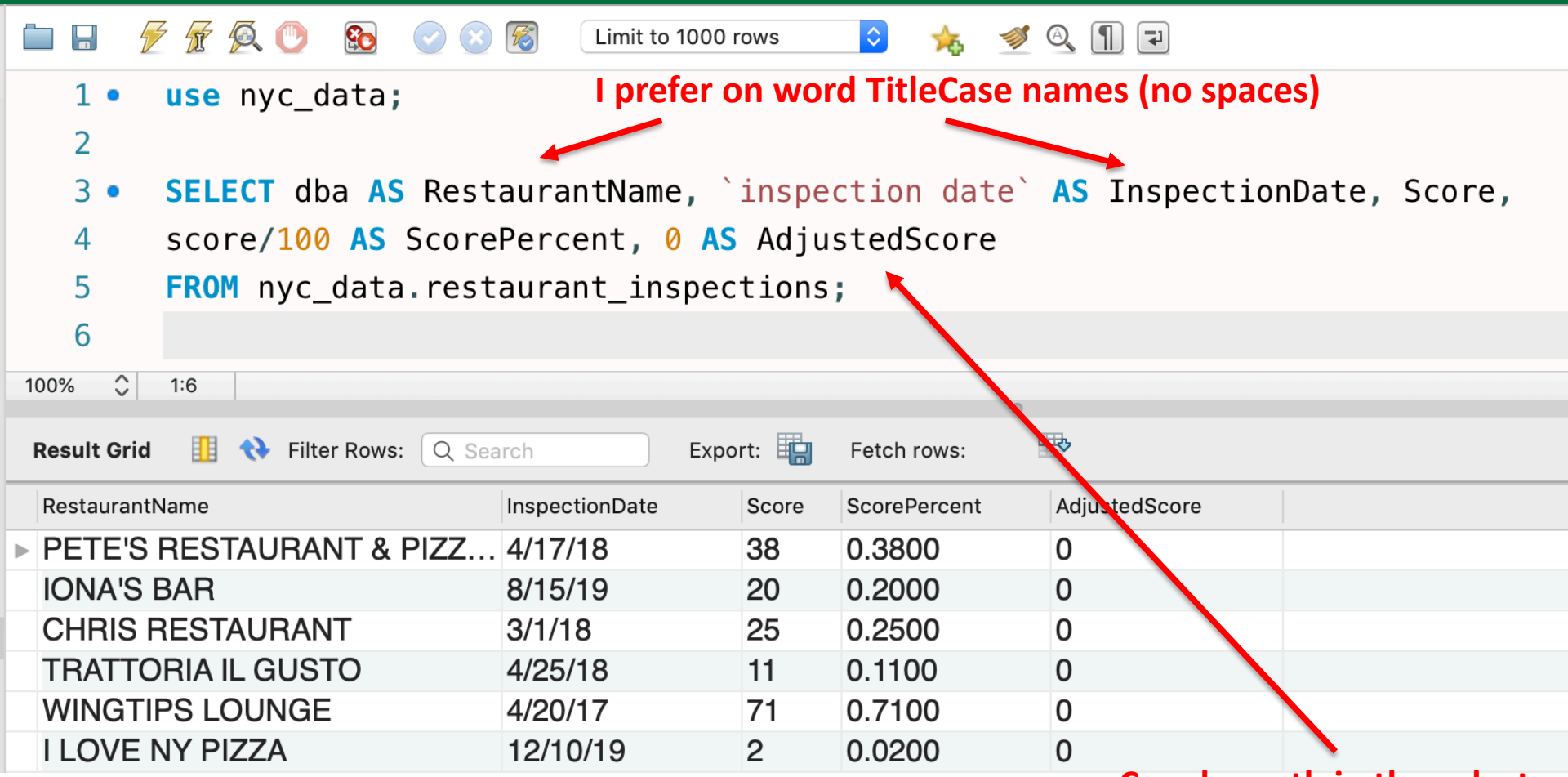
- Commands on multiple lines are ok (encouraged!)
- "where is not null" returns tuples where attribute is not null (can also say "is null" for nulls)
- "order by" sorts either asc (default) or desc
- Note the sorted DBA names start with '
- Also note lower case SQL commands are ok

Result Grid Filter Rows: Search Export: Fetch rows:

CAMIS	DBA	BORO	BUILDING	STREET	ZIPCODE	PHONE	CUISINE DESCRIPTI...	INSPECTION DATE	ACTION
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/12/19	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/28/17	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/28/17	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/28/17	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	4/3/18	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/29/18	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/28/17	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/12/19	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/28/17	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/29/18	Violations were cited in the followi
50064708	'CESCA	Manhattan	166	W 75TH ST	10023	2127586300	Italian	6/28/17	Violations were cited in the followi
50045928	'ESSEN	Manhattan	699	AVE OF A...	10010	2126330820	Delicatessen	9/13/19	Violations were cited in the followi
50045928	'ESSEN	Manhattan	699	AVE OF A...	10010	2126330820	Delicatessen	8/28/17	Violations were cited in the followi
50087274	'ESSEN	Manhattan	290	MADISO...	10017	2126890800	Delicatessen	1/29/19	Violations were cited in the followi
50045928	'ESSEN	Manhattan	699	AVE OF A...	10010	2126330820	Delicatessen	7/11/17	Violations were cited in the followi
50045928	'ESSEN	Manhattan	699	AVE OF A...	10010	2126330820	Delicatessen	7/11/17	Violations were cited in the followi

Source: https://data.cityofnewyork.us/Health/DOHMH-New-York-City-Restaurant-Inspection-Results/43nn-pn8j

Rename attributes (and tables) using the AS operator



Load this health inspection data into your local MySQL installation

1. Download “restaurant_inspections.sql” from today’s link on the course website Schedule page
2. Open MySQL Workbench
3. Connect to your localhost
4. Click File -> Open SQL Script…
5. Choose downloaded file from step 1
6. Run the script
7. This will create a database schema with one table holding all NYC restaurant health inspections (only for restaurants currently open)
8. Do exercises on next slide

Practice: use SELECT to answer questions about Ray's Pizza locations

Exercises

1. New Yorkers sometimes joke that there are many "Ray's Pizza" variants ("Original Ray's", "Famous Rays", "Famous Original Ray's"…), find all inspection for each the Ray's Pizza variants
   - Use "like" instead of = in a where clause (WHERE DBA LIKE 'Ray')
   - Like also works with wildcards
     - "%" matches all
     - "_" matches one character
2. What are the "gotchas" with Ray's name?
3. How many of Ray's are in the Bronx boro? Queens? Manhattan?
   - WHERE clauses can use "and", "or", "not"
4. Limit your results to only inspections that raised a "critical flag"
   - Use ` character (same key as tilde ~, by the 1 key) around attributes that have more than one word, e.g., `critical flag`
5. Would you eat at the Columbus Ave Ray's Pizza store?