CSE 199: UB Seminar (Fall 2026) - Data management

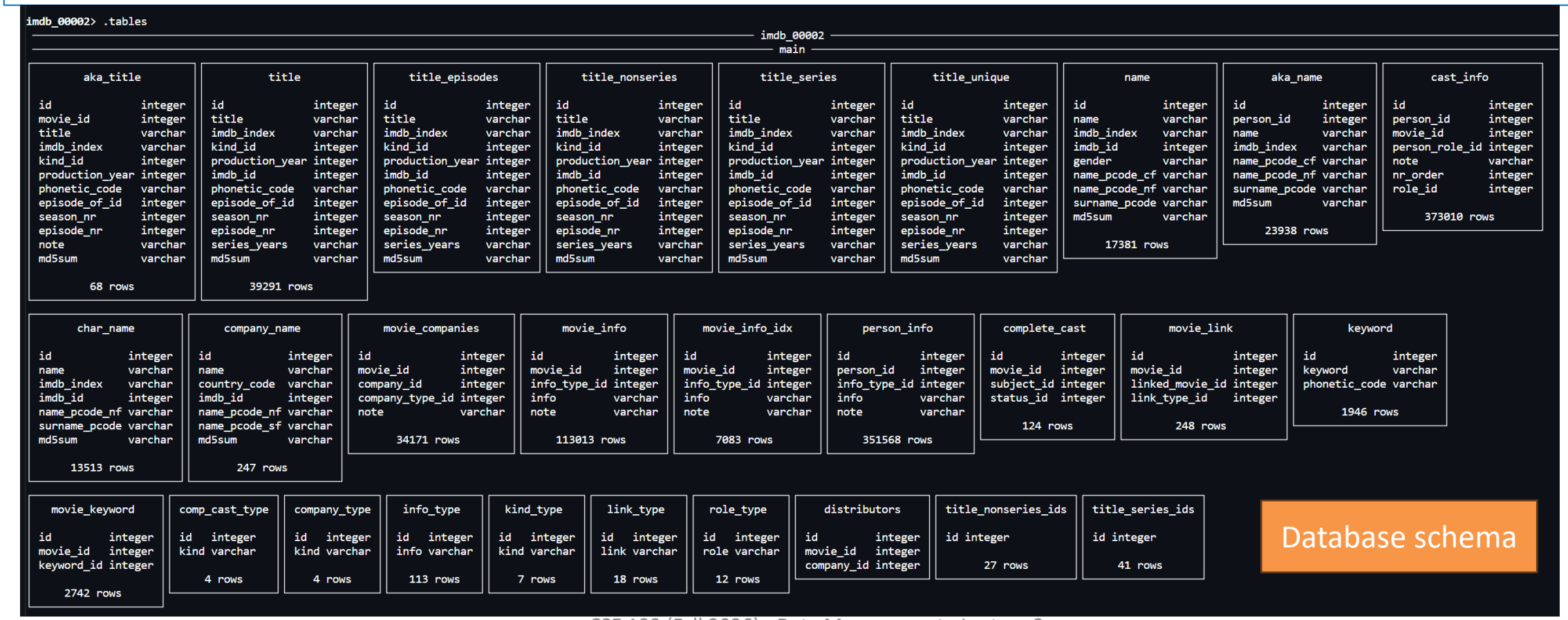
Lecture 3: Relational Model & SQL

Zhuoyue Zhao
zzhao35 [at] buffalo [dot] edu

University at Buffalo
Department of Computer Science and Engineering
School of Engineering and Applied Sciences

Relational Model

- Records => tuples (in mathematical sense)
- A collection of entities => set of tuples (relations) (in mathematical sense)



Database Instance

title	
id	integer
title	varchar
imdb_index	varchar
kind_id	integer
production_year	integer
imdb_id	integer
phonetic_code	varchar
episode_of_id	integer
season_nr	integer
episode_nr	integer
series_years	varchar
md5sum	varchar

39291 rows

Schema

Instance

id	title	imdb_index	kind_id	production_year	imdb_id	phonetic_code	episode_of_id	season_nr	episode_nr	series_years	md5sum
4703511	TV's All-Time Favorites Week 1 Game 4: Leave It to Beaver vs. Petticoat Junction		7			T1243	917574				f66c591fcfff451d2c22aaeb1343b03b
4710757	(2017-09-25)		7	2017			2635660				0a5f3556182f588a3ef5aa73d7cff650
4710754	(2017-09-22)		7	2017			2635660				c5457639932f6e5e41936433f14092eb
4710751	(2017-09-19)		7	2017			2635660				f338d55e4e5b88ef8b39068d75c365a8
4710750	(2017-09-18)		7	2017			2635660				f52a4327aab93f5e1641959da8339d3e
4710745	(2017-07-21)		7	2017			2635660				2e9a93ef5f0fad4afe3a7188230348df
4710746	(2017-07-22)		7	2017			2635660				2b88cd62d538815a6937eff45d9c20e9
4710748	(2017-09-16)		7	2017			2635660				1770f206f83df8dbacdbd01108a5d117
4710747	(2017-07-23)		7	2017			2635660				21c8de17b07a163ac6608802c6a66467
4710755	(2017-09-23)		7	2017			2635660				26d87158714b5f1e9cd80614d44404a6

Relational Query Languages

- Formal query languages
  - Relational algebra
    - Functional – describes how to query
  - Relational calculus
    - Declarative – describes what to query
  - No side effects! Does not include data definition, update, integrity checks, and etc.
  - Theoretical foundation of modern RDBMS; allows for query optimization

- Query language in practice: SQL (Structured Query Language)
  - Has its root in relational algebra and relational calculus
  - Includes many more beyond queries: imperative sublanguage, data definition, etc.

SQL Query Syntax

- `SELECT` and `FROM` clauses are mandatory
- `WHERE` clause is optional

```
SELECT  [DISTINCT] target-list
FROM    relation-list
[WHERE predicate]
```

- *`relation-list`*: a list of relation
  - each possibly with a table alias (aka correlation name)
- *`target-list`*: a list of expressions that may reference columns in the relation list
  - "`*`" to denote all the columns in the relation list
  - each may be renamed with `AS` clause (e.g., `S.name as student_name`)
  - `DISTINCT`: an optional keyword to deduplicate the result
- *`predicate`*: boolean expressions over the columns in the relation list, may contain
  - comparisons such as <, >, <=, >=, =, <>, LIKE
  - AND/OR/NOT
  - nested query
  - …

SQL supports string matching operator LIKE:
`_' stands for any one character and `%' stands for 0 or more arbitrary characters.
e.g., dname LIKE '%Engineering' will match all departments that ends with "Engineering" in its name

SQL Syntax Example

- Simple select without table.
  - Evaluates an expression (e.g., arithmetic, string operation, etc.)

```
imdb_00002> SELECT 1 + 1;
+---------+
| (1 + 1) |
+---------+
|       2 |
+---------+

imdb_00002> SELECT 'Buffalo is ' || 'in the state of NY';
+-----------------------------------------+
| ('Buffalo is ' || 'in the state of NY') |
+-----------------------------------------+
| Buffalo is in the state of NY           |
+-----------------------------------------+
```

SQL Syntax Example

- Listing all records in a table.

```
imdb_00002> SELECT * FROM kind_type;
```

id	kind
1	movie
2	tv series
3	tv movie
4	video movie
5	tv mini series
6	video game
7	episode

SQL Syntax Example

- Find the title named ‘At Home with Julia’

```
imdb_00002> select * from title where title = 'At Home with Julia';
```

id	title	imdb_index	kind_id	production_year	imdb_id	phonetic_code	episode_of_id	season_nr	episode_nr	series_years	md5sum
214700	At Home with Julia		2	2011		A3532				2011-2011	46d129939f7aa83aaa29141f9b972384

SQL Syntax Example

- Find the id and production year of the title ‘At Home with Julia’

```
imdb_00002> select id, production_year from title where title = 'At Home with Julia';
```

id	production_year
214700	2011

SQL Syntax Example

- Find the id and title (name) of the titles that contains keyword “Home”

```
imdb_00002> select id, title from title where title like '%Home%';
```

id	title
1704375	Home of Beika's Grenier
1704930	The Triplet's Country Home Murder Case
214700	At Home with Julia
2699117	Home
2635810	Bringin' Home the Bacon
2636245	Homemade Holiday Hits
2636244	Homemade Halloween
2636240	Home for the Holidays
2636241	Home for the Holidays - 2017
2635943	Down Home Delights
2636222	Healthy Homecooking
2699133	Long Way Home
2635959	Easy Homemade Holiday
2636037	Extra Value Friday: Homemade Holidays
2995342	Feeding the Home Team
2636246	Homemade with a Little Help
2636243	Homecoming Classics
2636242	Home Sweet Homemade
1554395	He Always Comes Home
1554540	The Night My Father Came Home
2995316	Bringin' Home the Farm
1205449	Spider-Man: Homecoming
1205389	Home Alone
1259890	Coming Home

SQL Syntax Example

- Find the id, title and production_year for any 5 titles produced on or after the year of 2015.

```
imdb_00002> select id, title, production_year from title where production_year >= 2015 limit 5;
```

id	title	production_year
4710757	(2017-09-25)	2017
4710754	(2017-09-22)	2017
4710751	(2017-09-19)	2017
4710750	(2017-09-18)	2017
4710745	(2017-07-21)	2017

SQL Syntax Example

- Find the id, title and production_year for any 5 movies produced on or after the year of 2015.

id	title	imdb_index	kind_id	p
4281131	Schaukellied		1	
3407369	Banned from Traveling		1	
3447176	Bluejackets for China		1	
3481419	Canned Ecology		1	
3566553	Deaf Dumb & Blind Date		1	
33460	6Teen		2	
45713	A Guy Walks Into a Bar		2	
519815	Como la vida		2	
501479	CNET CraveCast		2	
1704140	Meitantei Conan		2	

id	kind
1	movie
2	tv series
3	tv movie
4	video movie
5	tv mini series
6	video game
7	episode

A (equi-)join can connect two tables through keys. (though not always….)

SQL Syntax Example

- Find the id, title and production_year for any 5 movies produced on or after the year of 2015.

```
imdb_00002> select t.id, t.title, t.production_year
      ...> from title t, kind_type kt
      ...> where t.production_year >= 2015
      ...> and t.kind_id = kt.id and kt.kind = 'movie'
      ...> limit 5;
```

id	title	production_year
3407369	Banned from Traveling	2016
3566553	Deaf Dumb & Blind Date	2016
3800924	How to Be a Killer	2016
3647997	Emergent Part Two	2015
4683278	Yu Baosi	2016

Or

(they are equivalent)

```
imdb_00002> select t.id, t.title, t.production_year
      ...> from title t join kind_type kt on t.kind_id = kt.id
      ...> where t.production_year >= 2015
      ...> and kt.kind = 'movie'
      ...> limit 5;
```

id	title	production_year
3407369	Banned from Traveling	2016
3566553	Deaf Dumb & Blind Date	2016
3800924	How to Be a Killer	2016
3647997	Emergent Part Two	2015
4683278	Yu Baosi	2016

SQL Syntax Example

- Find the id, title and production_year for the 5 movies produced on or after the year of 2015 with the lowest/highest IDs.

```
imdb_00002> select t.id, t.title, t.production_year
       ...> from title t join kind_type kt on t.kind_id = kt.id
       ...> where t.production_year >= 2015
       ...> and kt.kind = 'movie'
       ...> order by t.id
       ...> limit 5;
```

id	title	production_year
3407369	Banned from Traveling	2016
3566553	Deaf Dumb & Blind Date	2016
3647997	Emergent Part Two	2015
3800924	How to Be a Killer	2016
4683278	Yu Baosi	2016

```
imdb_00002> select t.id, t.title, t.production_year
       ...> from title t join kind_type kt on t.kind_id = kt.id
       ...> where t.production_year >= 2015
       ...> and kt.kind = 'movie'
       ...> order by t.id desc
```

id	title	production_year
4683278	Yu Baosi	2016
3800924	How to Be a Killer	2016
3647997	Emergent Part Two	2015
3566553	Deaf Dumb & Blind Date	2016
3407369	Banned from Traveling	2016

SQL Syntax Example

- Count how many movies are there in the database?

```
imdb_00002> select count(*) from title t join kind_type kt on t.kind_id = kt.id
       ...> where  kt.kind = 'movie';
```

count_star()
19

- SQL also supports many other aggregation functions:
  - COUNT(*) – number of result rows
  - COUNT(expr) – number of non-null rows
  - MIN, MAX, SUM, AVG, VARIANCE, STDDEV

Quiz time

- Submit your answer in tophat