SQL Programming 1

Reading:

- ”The Database language SQL” chapter
- SQLite tutorial: https://www.sqlitetutorial.net/

Why SQL?

- SQL is a very-high-level language.
  - Say “what to do” rather than “how to do it.”
  - Avoid a lot of data-manipulation details needed in procedural languages like C++ or Java.
- Database management system figures out “best” way to execute a query.
  - Called ”query processing” and “query optimization”.

SELECT-FROM-WHERE

An SQL query consists of three components: a SELECT clause, a FROM clause and an optional WHERE clause .

SELECT desired attributes (output attributes)
FROM one or more tables (input tables)
WHERE conditions on tuples of tables (condition for rows)

Our Running Example

The Movies (Small) database:

Movie(mvID, Title, Rating, Rel_date, Length, Studio)

Classification(mvID*, Genre)

Cast(mvID*, Actor)

Direct(mvID*, Director)

All our SQL queries will be based on the Movies database.

Example

- What movies are produced by Roadshow?

```
SELECT title
FROM Movie
WHERE studio = 'Roadshow';
```

- The answer is a table with a single column, Title, and rows of the title for each movie by Roadshow.

```
TITLE
-----------------------------------------------
Coco Avant Chanel
Harry Potter and the Half-Blood Prince
```

The FROM-WHERE-SELECT Process

- FROM: Specify the input tables. Imagine a pointer visiting each row of the table in FROM.
- WHERE: Check if the “current” row satisfies the WHERE clause.
- SELECT: Specify the output columns. If so, compute the columns or expressions of the SELECT clause using the components of this row.

The FROM-WHERE-SELECT process ...

Movie

mvID	Title	Rating	Rel_date	Length	Studio
1	Angels & Demons	M	14-05-2009	138	Sony Pictures
2	Coco Avant Chanel	PG	25-06-2009	108	Roadshow
3	Harry Potter and the Half-Blood Prince	M	15-07-2009	153	Roadshow
4	The Proposal	PG	18-06-2009	107	Disney
5	Ice Age: Dawn of the Dinosaurs	PG	01-07-2009	94	20th Century Fox

Title
Coco Avant Chanel
Harry Potter and the Half-Blood Prince

Select title
From movie
Where studio='Roadshow';

SELECT *

- * in the SELECT clause stands for “all columns of tables in the FROM clause.”
- Example:

```
SELECT *
FROM Movie
WHERE studio = 'Roadshow';
```

Result of Query

Movie

mvID	Title	Rating	Rel_date	Length	Studio
1	Angels & Demons	M	14-05-2009	138	Sony Pictures
2	Coco Avant Chanel	PG	25-06-2009	108	Roadshow
3	Harry Potter and the Half-Blood Prince	M	15-07-2009	153	Roadshow
4	The Proposal	PG	18-06-2009	107	Disney
5	Ice Age: Dawn of the Dinosaurs	PG	01-07-2009	94	20th Century Fox

Movie

mvID	Title	Rating	Rel_date	Length	Studio
2	Coco Avant Chanel	PG	25-06-2009	108	Roadshow
3	Harry Potter and the Half-Blood Prince	M	15-07-2009	153	Roadshow

Now, the result has each column of the Movie table.

SELECT *Expressions*

- Any expression that makes sense can appear as an element of a SELECT clause.
- Example: Find the length of movies in hours.

```
SELECT title, length/60
FROM movie
```

TITLE	LENGTH/60
Angels and Demons	2.3
Coco Avant Chanel	1.8
Harry Potter and the Half-Blood Prince	2.55
The Proposal	1.78333333
Ice Age: Dawn of the Dinosaurs	1.56666667

Aliases: SELECT … AS …

- Columns can be renamed using AS …. This is often used for producing reports.

```
SELECT title, length/60 as LenInHours
FROM movie;
```

TITLE	LENGINHOURS
Angels and Demons	2.3
Coco Avant Chanel	1.8
Harry Potter and the Half-Blood Prince	2.55
The Proposal	1.78333333
Ice Age: Dawn of the Dinosaurs	1.56666667

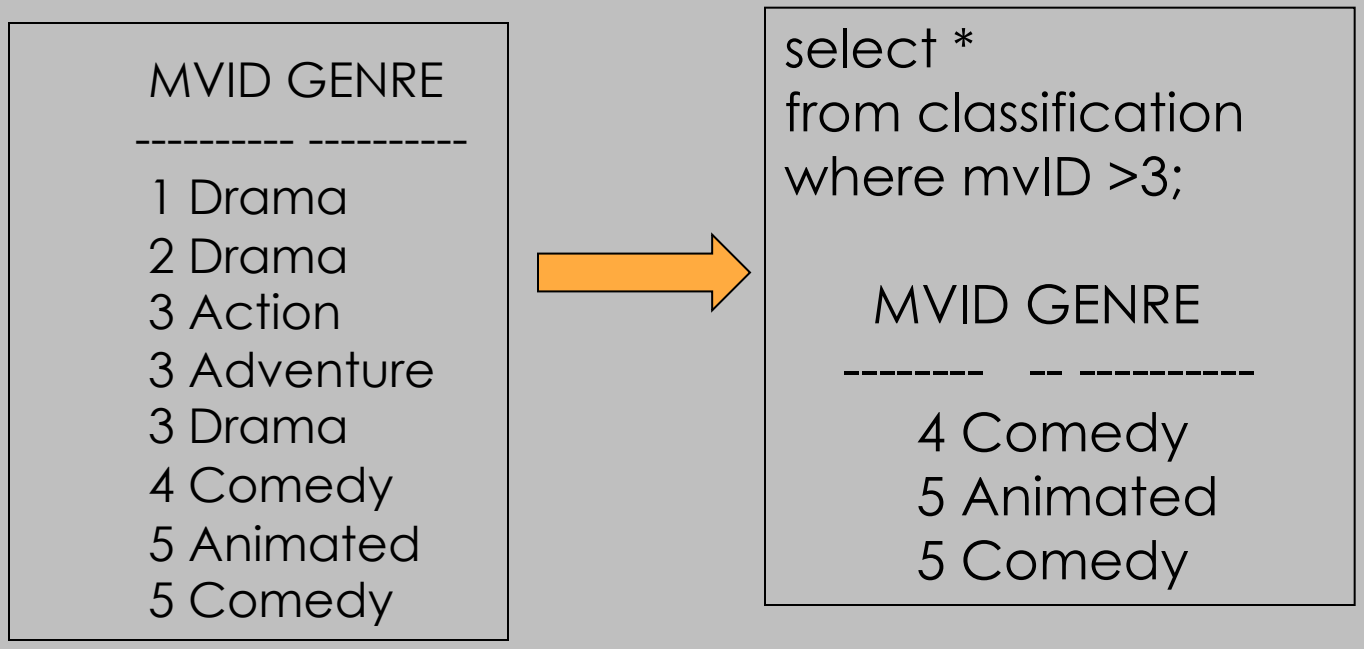
Select Constants

```
SELECT 'I like', title
FROM Movie
WHERE studio = 'Roadshow';
```

```
'ILIKE TITLE
------ ----------------------------------------------
I like Coco Avant Chanel
I like Harry Potter and the Half-Blood Prince
```

The WHERE Clause

- In the WHERE clause, an expression is evaluated to a boolean value (TRUE or FALSE). Tuples are selected only if the WHERE expression is evaluated to TRUE.
- Comparisons result in boolean values.
  - =, <>, <, >, <=, >=, !=.
- Example: Find the genre of movies whose mvID is >3.



The WHERE Clause ...

- Many other operators also result in boolean values. These more complex operators will be explained later.
  - LIKE, IS NULL, IS NOT NULL.
- Logical operators combine boolean expressions into a complex condition.
  - NOT, AND, OR.

Example: Complex Condition

- Find the title of movies produced by Roadshow and with rating ‘PG’.

```
SELECT title
FROM Movie
WHERE studio=‘Roadshow’ AND rating=‘PG’;
```

String Patterns for Comparison

- A condition can compare a string to a pattern by:
  - <Attribute> LIKE <pattern> or <Attribute> NOT LIKE <pattern>
- Pattern is a quoted string with
  - %: any string;
  - _: any character.

Example: LIKE

- Find the movies whose title has the string "Age". Return their title.

```
SELECT title
FROM Movie
WHERE title LIKE '%Age%';
```

NULL Values

- Tuples in SQL relations can have NULL as a value for one or more components.
- Meaning depends on context. Two common cases:
  - *Missing value* : e.g., we do not know the length of movie “Angels and Demons”.
  - *Inapplicable* : e.g., the value of attribute spouse for an unmarried person.

NULL Comparison

- WHERE … IS NULL:
  - Check if an attribute value equals to NULL.

Example: Find the movies where the production studio information is missing.

```
SELECT *
FROM Movie
WHERE studio IS NULL
```

NULL Comparison …

- WHERE … IS NOT NULL.
  - Check if an attribute does not take NULL value.

Exercise: Explain the following query in English.

```
SELECT *
FROM Movie
WHERE studio IS NOT NULL
```

Exercises

Write an SQL query to find the mvID of movies that have either "Tom Hanks" or "Audrey Tautou".

MVID	ACTOR
1	Tom Hanks
2	Alessandro Nivola
2	Audrey Tautou
2	Benolt Poelvoorde
2	Marie Gillain
3	Daniel Radcliffe
3	Emma Watson
3	Rupert Grint
4	Betty White
4	Malin Akerman
4	Mary Steenburgen
4	Ryan Reynolds
4	Sandra Bullock
5	Chris Wedge
5	Denis Leary
5	John Leguizamo
5	Queen Latifah
5	Ray Ramono

Exercises …

The query given below is meant to find the mvID of movies that have "Marie Gillain" and "Audrey Tautou". Will the query do the job? What's the result for the following queries?

MVID	ACTOR
1	Tom Hanks
2	Alessandro Nivola
2	Audrey Tautou
2	Benolt Poelvoorde
2	Marie Gillain
3	Daniel Radcliffe
3	Emma Watson
3	Rupert Grint
4	Betty White
4	Malin Akerman
4	Mary Steenburgen
4	Ryan Reynolds
4	Sandra Bullock
5	Chris Wedge
5	Denis Leary
5	John Leguizamo
5	Queen Latifah
5	Ray Ramono

```
select mvID
from cast
where actor='Marie Gillain'
        AND actor='Audrey Tautou';
```

Notes: The WHERE clause run across all rows in the input table. So there are not any grammar errors.

DISTINCT: Removing Duplicates

- What are the genres for movies?

  SELECT genre

  FROM Classification;
- Each genre is listed as many times as there are movie-genre pairs.

GENRE
Drama
Drama
Action
Adventure
Drama
Comedy
Animated
Comedy

DISTINCT: Removing Duplicates

- What are the (distinct) genres for movies?

  SELECT **DISTINCT** genre

  FROM Classification;
- Each different genre is listed only once.

GENRE
Adventure
Animated
Action
Comedy
Drama

ORDER BY: Ordering Output

- By default ORDER BY sorts output into ascending order. But can use ORDERY BY … DESC to sort into descending order.
- Example:

```
select mvID, title, length
from movie
order by length desc
```

MVID	TITLE	LENGTH
3	Harry Potter and the Half-Blood Prince	153
1	Angels and Demons	138
2	Coco Avant Chanel	108
4	The Proposal	107
5	Ice Age: Dawn of the Dinosaurs	94

Aggregation

- SUM, AVG, COUNT, MIN, and MAX can be applied to a column in a SELECT clause to produce aggregates for the column.
- Also, COUNT(*) counts the number of tuples.

Example: Aggregation

- Overall aggregation: Aggregate all tuples in a table.
- Example: What's the average length of movies overall?

```
SELECT AVG(length)
FROM Movie;
```

MVID	LENGTH
1	138
2	108
3	153
4	107
5	94

AVG(LENGTH)
120

Eliminating Duplicates in an Aggregation

- Use DISTINCT inside an aggregation.
- Example: How many studios are there in total?

```
SELECT COUNT(DISTINCT studio)
FROM Movie;
```

Selective Aggregation

- Use WHERE condition to first select tuples and then aggregate them.
- Example: How many studios are there where their movie length>100 minutes?

```
SELECT COUNT(DISTINCT studio)
FROM Movie
Where length>100;
```

Join Queries

- Interesting queries often combine data from more than one relation.
  - Join tables -- list more than one table in the FROM part:

    SELECT *

    FROM Movie, Direct
- No JOIN conditions produce the Cartesian Product of two tables.
  - All combinations of rows in each table.

The Cartesian Product

MVID	TITLE	MVID	DIRECTOR
1	Angels and Demons	1	Ron Howard
1	Angels and Demons	2	Anne Fontaine
1	Angels and Demons	3	David Yates
1	Angels and Demons	4	Anne Fletcher
1	Angels and Demons	5	Carlos Saldanha
1	Angels and Demons	5	Mike Thurmeier
2	Coco Avant Chanel	1	Ron Howard
2	Coco Avant Chanel	2	Anne Fontaine
2	Coco Avant Chanel	3	David Yates
2	Coco Avant Chanel	4	Anne Fletcher
2	Coco Avant Chanel	5	Carlos Saldanha
2	Coco Avant Chanel	5	Mike Thurmeier
3	Harry Potter and the Half-Blood Prince	1	Ron Howard
3	Harry Potter and the Half-Blood Prince	2	Anne Fontaine
3	Harry Potter and the Half-Blood Prince	3	David Yates
3	Harry Potter and the Half-Blood Prince	4	Anne Fletcher
3	Harry Potter and the Half-Blood Prince	5	Carlos Saldanha
3	Harry Potter and the Half-Blood Prince	5	Mike Thurmeier
4	The Proposal	1	Ron Howard
4	The Proposal	2	Anne Fontaine
4	The Proposal	3	David Yates
4	The Proposal	4	Anne Fletcher
4	The Proposal	5	Carlos Saldanha
4	The Proposal	5	Mike Thurmeier
5	Ice Age: Dawn of the Dinosaurs	1	Ron Howard
5	Ice Age: Dawn of the Dinosaurs	2	Anne Fontaine
5	Ice Age: Dawn of the Dinosaurs	3	David Yates
5	Ice Age: Dawn of the Dinosaurs	4	Anne Fletcher
5	Ice Age: Dawn of the Dinosaurs	5	Carlos Saldanha
5	Ice Age: Dawn of the Dinosaurs	5	Mike Thurmeier

There are a total of 5*6=30 rows in the result.

There are a lot of rows. But the information of directors for movies is lost.

Join Queries: the WHERE part

It is ESSENTIAL that you have a WHERE part in a join query:

- List all tables in the FROM part.
- Specify the JOIN condition in the WHERE part.
- Specify the attributes to output in the SELECT part.

```
SELECT Movie.mvID, Movie.title, Direct.mvID, Direct.director
FROM Movie, Direct
WHERE Movie.mvID=Direct.mvID.
```

Join Queries ...

MVID	TITLE
1	Angels and Demons...
2	Coco Avant Chane…
3	Harry Potter and the …
4	The Proposal …
5	Ice Age: Dawn of the ...

MVID	DIRECTOR
1	Ron Howard
2	Anne Fontaine
3	David Yates
4	Anne Fletcher
5	Carlos Saldanha
5	Mike Thurmeier

SELECT Movie.mvID, Movie.title, Direct.mvID, Direct.director
FROM Movie, Direct
WHERE Movie.mvID=Direct.mvID.

	mvID	title	mvID:1	director
1	1	Angels & Demons	1	Ron Howard
2	2	Coco Avant Chanel	2	Annie Fontaine
3	3	Harry Potter and the Half-Blood Prince	3	David Yales
4	4	The Proposal	4	Anne Fletcher
5	5	Ice Age: Dawn of the Dinosaurs	5	Carlos Saldanha
6	5	Ice Age: Dawn of the Dinosaurs	5	Mike Thurmeier

Join Queries …

Explain the below query in English.

```
SELECT Movie.mvID, Movie.title, Direct.mvID, Direct.director
FROM Movie, Direct
WHERE Movie.mvID=Direct.mvID
        and Direct.director='Ron Howard';
```

Summary

- An SQL query structure:

SELECT *output attributes or aggregates*
FROM *list of tables*
WHERE *conditions for selecting tuples*
ORDER BY *attributes for ordering output*