MySQL intro

Glossary

- *Schema* - database, main storage of data
- *Table* - contains information about entity and its attributes.
- *Query* - question that you can ask of the data stored in database.
- *Views* - saved queries.

Data Types

Numeric Data Types:

- **INT**: Integer up to 10 digits.
- **BIGINT**: Integer up to 19 digits.
- **FLOAT**: Floating decimal point number.

Text Data Types:

- **CHAR**: Fixed-length string with fixed length (up to 255) specified in parentheses.
- **VARCHAR**: Fixed-length string with max length (up to 255) specified in parentheses.
- **TEXT**: String with a max length of 65,535 characters.

Date Data Types:

- **DATE**: Date formatted as YYYY-MM-DD.
- **DATETIME**: Date and time formatted as YYYY-MM-DD HH:MI:SS.
- **TIME**: Time formatted as HH:MI:SS.

Creating, Changing and Deleting Data

- **SELECT** - extracts data from a database
- **UPDATE** - updates data in a database
- **DELETE** - deletes data from a database
- **INSERT INTO** - inserts new data into a database
- **CREATE DATABASE** - creates a new database
- **ALTER DATABASE** - modifies a database
- **CREATE TABLE** - creates a new table
- **ALTER TABLE** - modifies a table
- **DROP TABLE** - deletes a table
- **CREATE INDEX** - creates an index (search key)
- **DROP INDEX** - deletes an index

Example

```
create table entity(
    entity_id float(10, 2) not null
    , unit_id float(10, 2) not null
    , some_state varchar(100)
    -- ...other important fields
    , primary key(entity_id)
);
```

Querying Data

- **SELECT**: Allows you to choose the specific fields you would like displayed in your query results.
- **FROM**: Allows you to specify what table in the database the data is going to come from.
- **WHERE**: Allows you to specify some conditions to filter the data.
- **ORDER BY**: Allows you to sort the data in ascending or descending order by multiple fields.
- **GROUP BY**: Allows you to specify by which fields you want to group your data.
- **LIMIT:** Limits the output by certain number of first rows.

Example

```
Select *
From ratings;
```

```
Select AppName, Price, TotalRatings, OverallRating, Genre
From ratings
Where Price = 0 and OverallRating >= 4
Order by TotalRatings Desc
Limit 20;
```

General case

```
Select column_name1, column_name2, …,column_name_last
From table_name
Where condition1 and condition2 and …,condition_last
Order by rule [Desc]
Limit number_of_rows;
```

Database insights

Entities and attributes

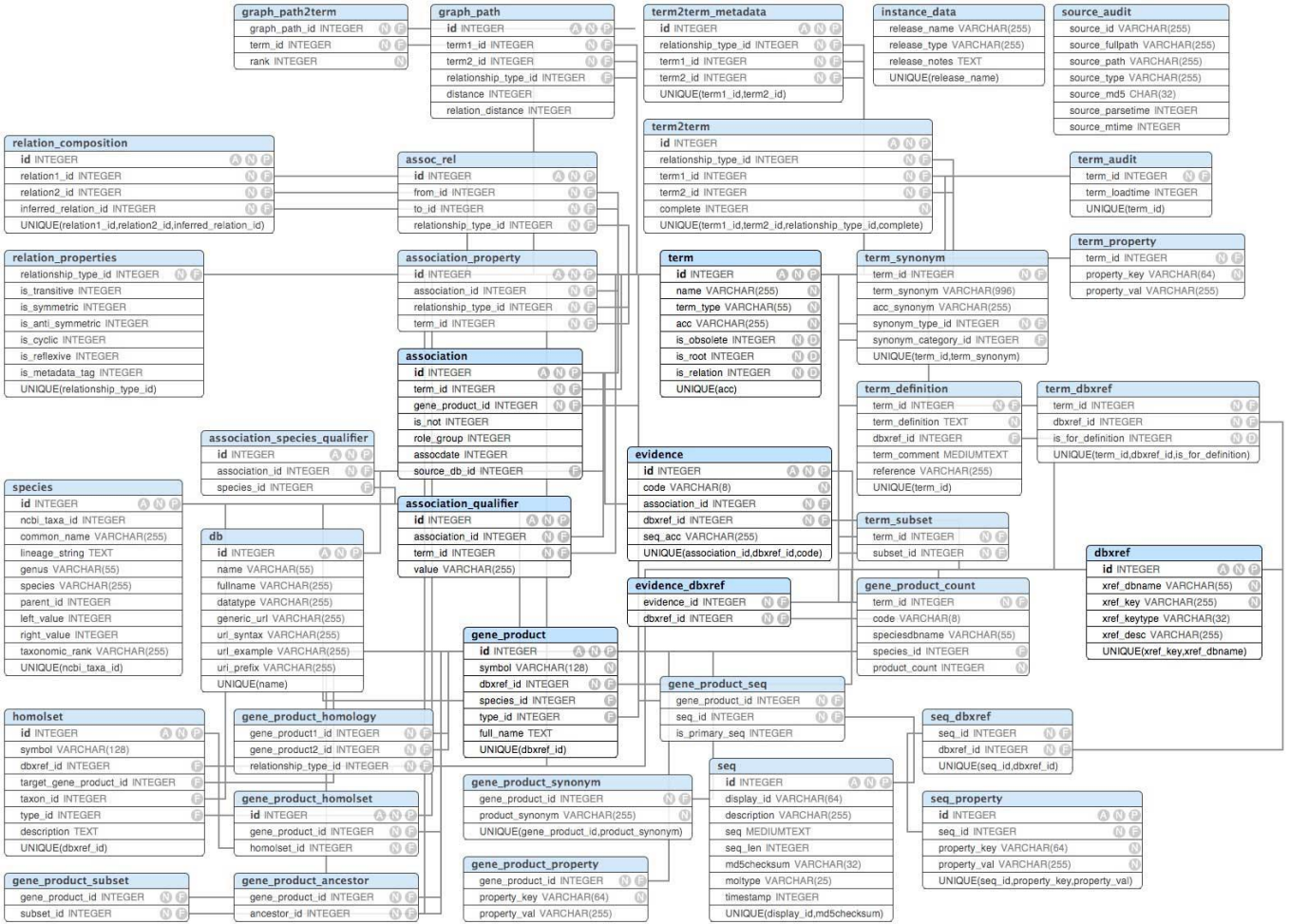
Id	Bookid	Memberid	Borrowdate	Returndate
1	1	3	01-20-2016	03-17-2016
2	2	4	01-19-2016	03-23-2016
3	1	1	02-17-2016	05-18-2016
4	4	2	12-15-2015	04-13-2016
5	2	2	02-18-2016	04-19-2016
6	3	5	02-29-2016	04-11-2016
Borrowers				

Bookid	Title	Author	Published	Stock
1	Scion of Ikshvaku	Amish Tripathi	06-22-2015	2
2	The Lost Symbol	Dan Brown	07-22-2010	3
3	Who Will Cry When You Die?	Robin Sharma	06-15-2006	4
4	Inferno	Dan Brown	05-05-2014	3
5	The Fault in our Stars	John Green	01-03-2015	3
Books				

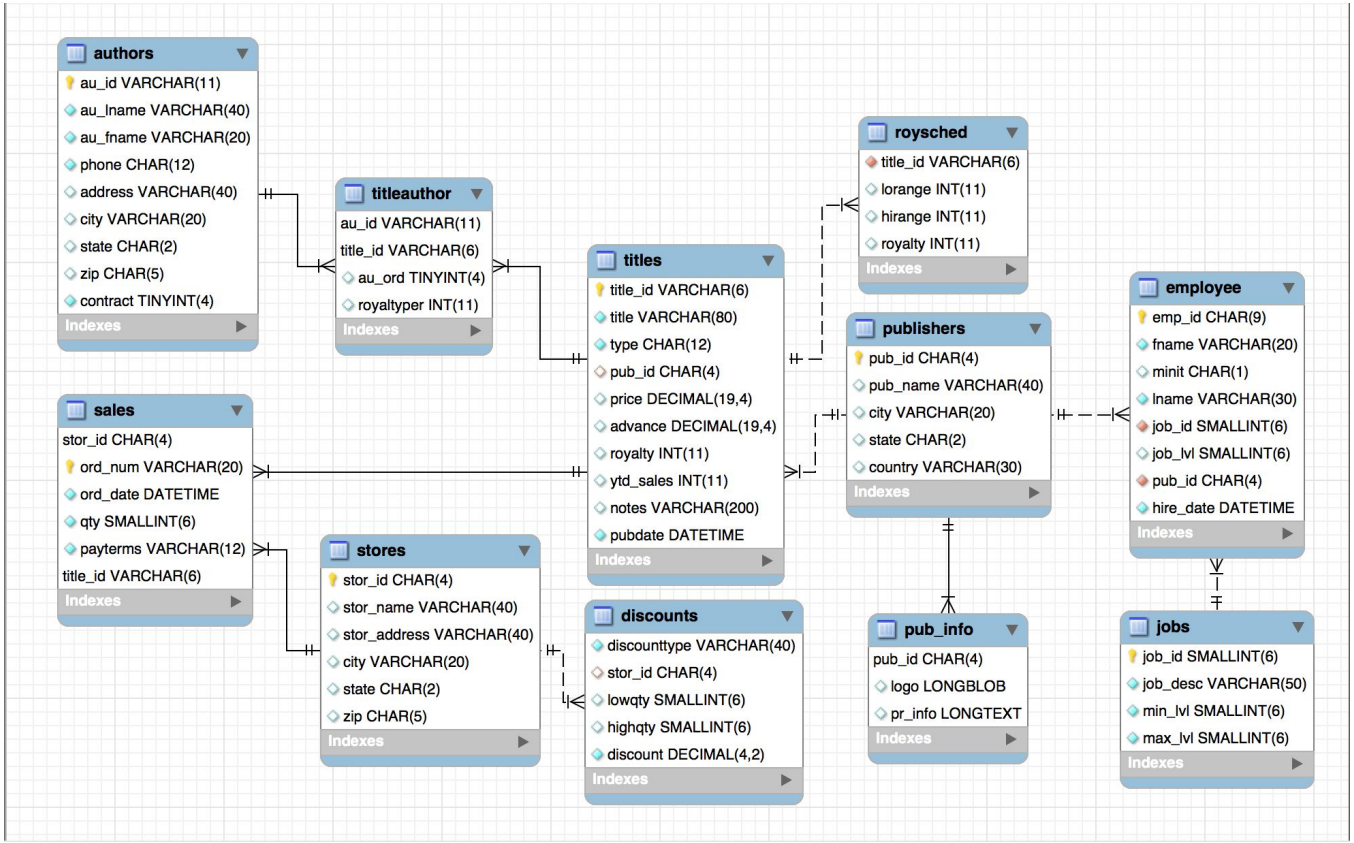
Memberid	Firstname	Lastname
1	Sue	Mason
2	Ellen	Horton
3	Henry	Clarke
4	Mike	Willis
5	Lida	Tyler
Members		

Relationships

- One-to-One
- One-to-Many
- Many-to-Many



Publications database



Labs Time

Joins and Relationships

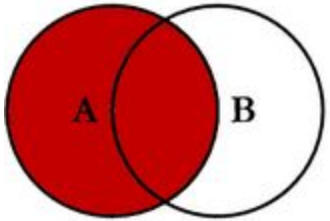
Instructions to get the data

1. Download publications.sql.zip. (https://bit.ly/2BlWObF)
2. Unzip the downloaded file and extract the database dump file on your machine.
3. In MySQL Workbench / Sequel Pro, create a new database called "publications" by running the following command:
   ```
   CREATE DATABASE publications;
   ```
4. Select Database by running the following command: `use publications;`
5. Import data:
   a. Mac: From the top menu, select File and then the Import menu option. Navigate to the publications.sql dump file and import it.
   b. Windows: Just open sql file and run it.

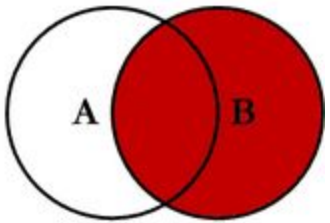
Joins

- INNER JOIN - only those which exist in both tables
- LEFT JOIN - all the records from left table + those who matched in right one
- RIGHT JOIN - all the records from right table + those who matched in left one
- OUTER JOIN - all records from both tables

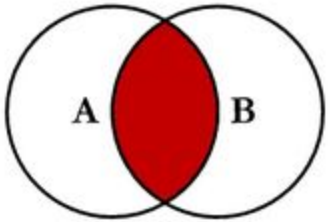
SQL JOINS



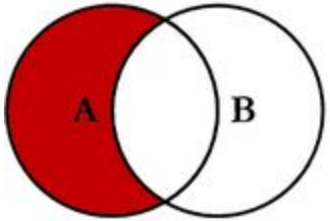

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



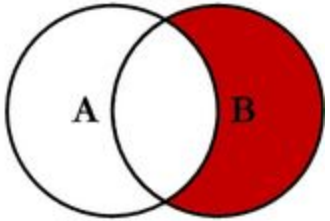

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



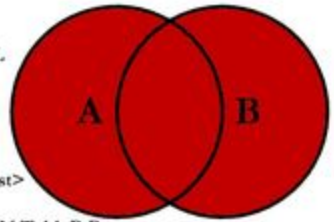

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



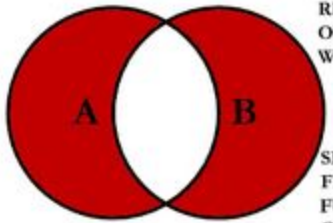

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

Examples

```
Select pubs.pub_name, COUNT(titles.title_id) AS Titles
From publications.publishers pubs
LEFT JOIN publications.titles titles
ON pubs.pub_id = titles.pub_id
GROUP BY pubs.pub_name;
```

```
Select *
From publications.employee emp
LEFT JOIN publications.jobs job
ON emp.job_id = job.job_id
UNION
Select *
From publications.employee emp
RIGHT JOIN publications.jobs job
ON emp.job_id = job.job_id;
```

Labs Time