Practical 7: Hands on SQL

Activity1: Create Database and Table using SQL

1. Start the MySQL connection you have created in Practical 4.
2. Create database called “egt206_prac7DB” and select it.

```
CREATE DATABASE egt206_prac7DB;
USE egt206_prac7DB;
```

3. Create four tables as following.

Tablename: books

id	title	booktype	author_id	editor_id	translator_id
1	Time to Grow Up!	original	11	21	
2	Your Trip	translated	15	22	32
3	Lovely Love	original	14	24	
4	Dream Your Life	original	11	24	
5	Oranges	translated	12	25	31
6	Your Happy Life	translated	15	22	33
7	Applied AI	translated	13	23	34
8	My Last Book	original	11	28	

Tablename: authors

id	first_name	last_name
11	Ellen	Writer
12	Olga	Savelleva
13	Jack	Smart
14	Donald	Brain
15	Yao	Dou

```sql
CREATE TABLE books
(
id int NOT NULL,
title varchar(100) NOT NULL,
booktype varchar(50) NOT NULL,
author_id int NOT NULL,
editor_id int NOT NULL,
translator_id int NULL,
primary key(id)
);

INSERT INTO books
VALUES (1,'Time to Grow Up!','original',11,21,NULL), (2,'Your Trip','translated',15,22,32),
(3,'Lovely Love','original',14,24,NULL), (4,'Dream Your Life','original',11,24,NULL),
(5,'Oranges','translated',12,25,31), (6,'Your Happy Life','translated',15,22,33),
(7,'Applied AI','translated',13,23,34), (8,'My Last Book','original',11,28,NULL);
```

```sql
CREATE TABLE authors
(
id int NOT NULL,
first_name varchar(50) NOT NULL,
last_name varchar (50)NOT NULL,
primary key(id)
);

INSERT INTO authors
VALUES (11,'Ellen','Writer'),(12,'Olga','Savelleva'),(13,'Jack','Smart'),(14,'Donald','Brain'),(15,'Yao','Dou');
```

```sql
CREATE TABLE editors
(
id int NOT NULL,
first_name varchar(50) NOT NULL,
last_name varchar (50)NOT NULL,
primary key(id)
);

INSERT INTO editors
VALUES (21,'Daniel','Brown'),(22,'Mark','Johnson'),(23,'Maria','Evans'),(24,'Cathrine','Roberts'),
(25,'Sebastian','Wright'),(26,'Barbara','Jones'),(27,'Matthew','Smith');
```

```sql
CREATE TABLE translators
(
id int NOT NULL,
first_name varchar(50) NOT NULL,
last_name varchar (50)NOT NULL,
primary key(id)
);

INSERT INTO translators
VALUES (31,'Ira','Davies'),(32,'Ling','Weng'),(33,'Kristian','Green'),(34,'Roman','Edwards');
```

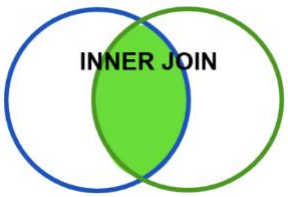
Tablename: editors

id	first_name	last_name
21	Daniel	Brown
22	Mark	Johnson
23	Maria	Evans
24	Cathrine	Roberts
25	Sebastian	Wright
26	Barbara	Jones
27	Matthew	Smith

Tablename: translators

id	first_name	last_name
31	Ira	Davies
32	Ling	Weng
33	Kristian	Green
34	Roman	Edwards

Activity2: Inner Joins / Join




```
SELECT *
FROM <tablename> AS A
INNER JOIN <tablename> AS B
ON <condition>;
```

The ON keyword is used together with INNER JOIN (or JOIN for short) for filtering of rows. It acts like a WHERE.

1. Display book id, titles along with their authors (i.e., the author's first name and last name).

```
SELECT a,id, a.title, b.first_name, b.last_name
FROM books AS a
INNER JOIN authors AS b
ON a.author_id = b.id;
```

2. Display book id, titles along with their translators (i.e., the translator's first and last name).

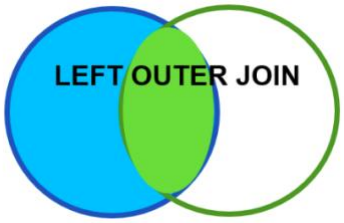
```
SELECT a.id, a.title, b.first_name, b.last_name
FROM books AS a
INNER JOIN translators AS b
ON a.translator_id = b.id;
```

Exercise

1. Display book id, titles along with their editors (i.e., the editor's first and last name).

-
```
SELECT a.id, a.title, b.first_name, b.last_name
FROM books AS a
INNER JOIN editors AS b
ON a.editor_id = b.id;
```

Activity3: Left Outer Join




```
SELECT *
FROM <tablename> AS A
LEFT OUTER JOIN <tablename> AS B
ON <condition>;
```

You should apply this when you want to keep all records from the left table and only the matched records from the right table.

1. Display book id, titles along with their authors and translators (taking only their last name). To get all of this data, we'll need to join three tables: books for basic info on the books, authors for the authors' last names, and translators for the translators' last names.

```
SELECT a.id, a.title, a.booktype, b.last_name AS author, c.last_name  AS translator
FROM books AS a
LEFT OUTER JOIN authors AS b
ON a.author_id = b.id
LEFT OUTER JOIN translators AS c
ON a.translator_id = c.id;
```

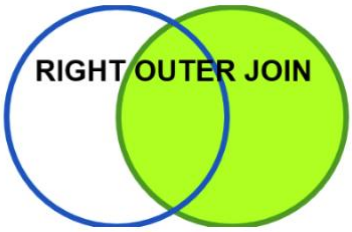
-
```
SELECT a.id, a.title, b.first_name, b.last_name
FROM books AS a
LEFT OUTER JOIN editors AS b
ON a.editor_id = b.id;
```

Exercise

1. Display book id, titles along with their editors (i.e., the editor's first and last name).

2. Compare the exercise's output between Activity 2 and Activity 3. Did you notice the difference between Inner Join and Left Outer Join? What is it?

Activity4: Right Outer Join




```
SELECT *
FROM <tablename> AS A
RIGHT OUTER JOIN <tablename> AS B
ON <condition>;
```

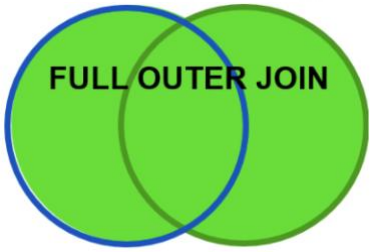
Right Outer Join is very similar to Left Outer Join. The only difference is that Right Outer Join keeps all records from the right table and only the matched records from the left table.

Exercise

1. Display all editors (i.e., the editor's first and last name) along with the book they are in-charge of.

```
SELECT a.id, a.title, b.first_name, b.last_name
FROM books AS a
RIGHT OUTER JOIN editors AS b
ON a.editor_id = b.id;
```

Activity5: Full Outer Join




```
SELECT *
FROM <tablename> AS A
LEFT OUTER JOIN <tablename> AS B
ON <condition>
UNION
SELECT *
FROM <tablename> AS A
RIGHT OUTER JOIN <tablename> AS B
ON <condition>
```

We use Full Outer Join when we want to keep all records from all tables, even unmatched ones. We do it by combining Left Outer Join and Right Outer Join using UNION.

Exercise

1. Display information about all of the books (id, title, booktype) and all of the editors (first name and last name) in one table.

```
SELECT a.id, a.title, a.booktype, b.first_name, b.last_name
FROM books AS a
LEFT OUTER JOIN editors AS b
ON a.editor_id = b.id
UNION
SELECT a.id, a.title, a.booktype, b.first_name, b.last_name
FROM books AS a
RIGHT OUTER JOIN editors AS b
ON a.editor_id = b.id;
```