



**Temasek Junior College**  
**2023 JC2 H2 Computing**  
**Database 4 – Writing SQL Statements**

### **Syllabus Objectives**

After completing this set of notes, you should be able to:

- Use DB Browser for SQLite to select record(s) using specified criteria
- Select data in the database using the SELECT statement
- Understand the concepts of inner join and left outer join
- Update records in database using UPDATE statement
- Insert records in database using INSERT INTO statement
- Delete records from database using DELETE statement
- Create table from database using CREATE TABLE statement
- Understanding the result of using AUTOINCREMENT in creating tables
- Delete table from database using DROP TABLE statement
- Use arithmetic, comparison and logical operators in SQL statements
- Use aggregate functions MIN, MAX, SUM and COUNT

## **1 Introduction**

Recall that a database is a set of data stored on one or more computers or servers.

Take as an example, a librarian has keyed in the book and borrower details given in the previous lessons.

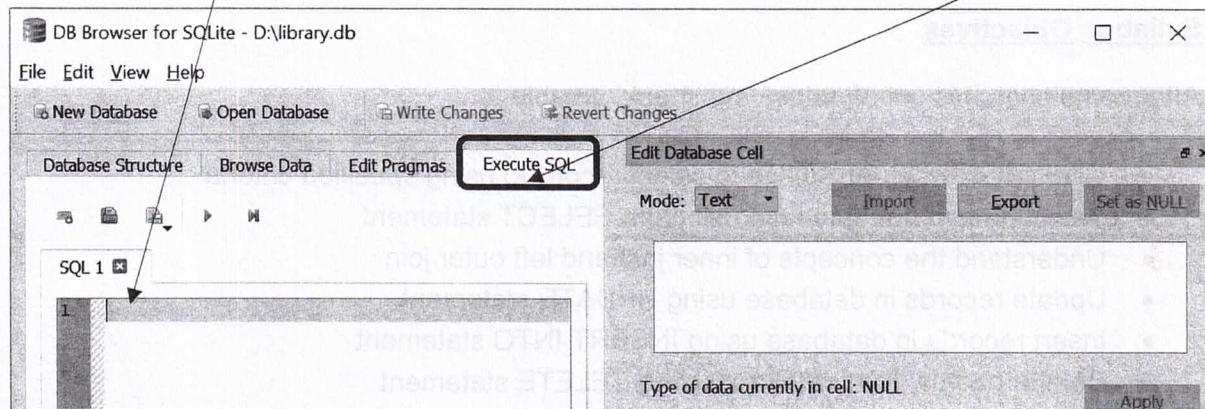
The book details are stored in Book table, with the publisher details stored in Publisher table, the borrower's details in Borrower table and the loans made in Loan table. The intention is to be able to retrieve, edit or delete the data easily.

This can be done through Data Manipulation Language (DML) statements like SELECT, INSERT, UPDATE and DELETE.

In this lesson, we will learn the SQL syntax to run the various queries in the database.

## Entering SQL into DB Browser for SQLite

To enter SQL into DB Browser, after loading the database, click on the **Execute SQL** tab. There is a text area for you to type in your SQL commands.



You can also click on the buttons above the text area.

|  |  |
|--|--|
|  | Creates a new tab for entering SQL.                        |
|  | Loads SQL file   |
|  | Saves SQL entered as a text file (You may save it as .sql) |
|  | Execute all SQL statements in that tab.                    |
|  | Execute the current line only.                             |

After executing SQL commands or making changes, you can click the following:

|  |   |
|--|---|
|  | Save changes made into the file                           |
|  | Undo changes made to the database, i.e. reload from file. |

Let us now look at the SQL statements that you can enter.

The database used in the following statements uses library.db.  
You can load the database into SQLite.

## SELECT

The SELECT statement allows the user to retrieve data from the database.

To select all fields, use \*.

For example, typing

```
SELECT * FROM Book
```

and clicking ► will give you details of all the books in library.

| ID | Title                       | PublisherID | Damaged |
|----|-----------------------------|-------------|---------|
| 1  | The Lone Gatsby             | 5           | 0       |
| 2  | A Winter's Slumber          | 4           | 1       |
| 3  | Life of Pie                 | 4           | 0       |
| 4  | A Brief History Of Primates | 3           | 0       |
| 5  | To Praise a Mocking Bird    | 2           | 0       |
| 6  | The Catcher in the Eye      | 1           | 1       |

Conditions may be added using WHERE.

For example, `SELECT * from Book WHERE Damaged = 1`  
The statement will return all the damaged books.

You can also find NULL values, for example books with no publishers (no value on the PublisherID field), using the IS operator.

```
SELECT * from Book WHERE PublisherID IS NULL
```

You can also search for terms which are not NULL, for example:

```
SELECT * from Book WHERE PublisherID IS NOT NULL
```

This statement returns all records where there is a publisher.

You can insert more than one condition using AND and OR binary operators.

For example,

```
SELECT * FROM Book WHERE Title = 'Life of Pie' AND Damaged = 0
```

This statement returns the books with Title 'Life of Pie' **and** are not damaged.

What if you only need the book titles?

In that case, include the fields you want in the SELECT statement. For example,

```
SELECT Title FROM Book
```

This statement will give you all the Title of the books in the Book table.

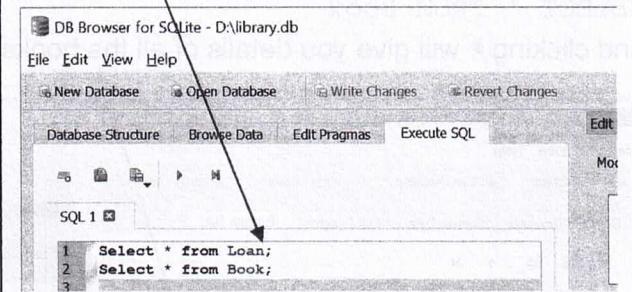
### Title

The Lone Gatsby  
A Winter's Slumber  
Life of Pie  
A Brief History Of Primates  
To Praise a Mocking Bird  
The Catcher in the Eye  
H2 Computing Ten Year Series

Try out the SQL statements yourself!

### Do you know?

You can execute multiple SQL statements, but each statement must end with a colon ;



### Quiz

- Which of the following SQL statements show all data inside the Publisher table?  
(Tick the statements)

|   |   |
|---|---|
|   | Select all FROM Publisher                             |
| ✓ | Select * FROM Publisher                               |
|   | Select ID, Title FROM Publisher                       |
| ✓ | Select ID, Title, PublisherID, Damaged FROM Publisher |

- Write down the SQL statement to show all the Names of the publishers inside the Publisher table.

SELECT Name FROM Publisher

- What does the following SQL statement do?

SELECT Title FROM Book WHERE PublisherID=1 AND Damaged=0

Displays the titles of books where publisherID = 1 and the book is not damaged

- Write down the SQL statement to show all the Titles of the books which have PublisherID 1 or 2.

SELECT Title FROM Book WHERE PublisherID=1 or PublisherID=2

Another example is to look at all the loans.

To list all loans, the SQL statement `SELECT * FROM Loan` is used.

| ID | BorrowerID | BookID | Date Borrowed |
|----|------------|--------|---------------|
| 1  | 3          | 2      | 20180220      |
| 2  | 3          | 1      | 20171215      |
| 3  | 2          | 3      | 20171231      |
| 4  | 1          | 5      | 20180111      |

The list can be ordered by the BookID (in ascending order) instead by using the SQL statement `SELECT * FROM Loan ORDER BY BookID ASC`

The result is as follows.

| ID | BorrowerID | BookID | Date Borrowed |
|----|------------|--------|---------------|
| 2  | 3          | 1      | 20171215      |
| 1  | 3          | 2      | 20180220      |
| 3  | 2          | 3      | 20171231      |
| 4  | 1          | 5      | 20180111      |

The list can be ordered by the BookID (in descending order) by using the SQL statement `SELECT * FROM Loan ORDER BY BookID DESC`

The result is as follows.

| ID | BorrowerID | BookID | Date Borrowed |
|----|------------|--------|---------------|
| 4  | 1          | 5      | 20180111      |
| 3  | 2          | 3      | 20171231      |
| 1  | 3          | 2      | 20180220      |
| 2  | 3          | 1      | 20171215      |

What happens if the following SQL statement is executed?

`SELECT * FROM Loan ORDER BY BookID`

By default, it will order by BookID in ascending order.

Try out the SQL statements using DBViewer in SQLite.

### Quiz

1. Write down the SQL statement to select all the records in Loan table arranged in ascending order of BookID with BorrowerID = 1.

SELECT \* FROM Loan WHERE BorrowerID=1 ORDER BY BookID=1 ASC

WHERE → ORDER

What if you want to find out the total number of loans?

You can use function COUNT to get the answer. The SQL can be the following:

```
SELECT COUNT(*) FROM Loan
```

---

## SQL JOIN

A SQL JOIN allows the combination of data from two sets of data (i.e. two tables). There are different types of joins. We look at three: **cross join**, **inner join** and **left outer join**.

Cross join returns the Cartesian product of rows from the tables in the join. It combines each row in the first table with each row in the second table.

For example, the following SQL statement

```
SELECT * FROM Book, Publisher
```

produces the following table.

| ID | Title                       | PublisherID | Damaged | ID | Name         |
|----|-----------------------------|-------------|---------|----|--------------|
| 1  | The Lone Gatsby             | 5           | 0       | 1  | NPH          |
| 1  | The Lone Gatsby             | 5           | 0       | 2  | Unpop        |
| 1  | The Lone Gatsby             | 5           | 0       | 3  | Appleson     |
| 1  | The Lone Gatsby             | 5           | 0       | 4  | Squirrel     |
| 1  | The Lone Gatsby             | 5           | 0       | 5  | Yellow Flame |
| 2  | A Winter's Slumber          | 4           | 1       | 1  | NPH          |
| 2  | A Winter's Slumber          | 4           | 1       | 2  | Unpop        |
| 2  | A Winter's Slumber          | 4           | 1       | 3  | Appleson     |
| 2  | A Winter's Slumber          | 4           | 1       | 4  | Squirrel     |
| 2  | A Winter's Slumber          | 4           | 1       | 5  | Yellow Flame |
| 3  | Life of Pie                 | 4           | 0       | 1  | NPH          |
| 3  | Life of Pie                 | 4           | 0       | 2  | Unpop        |
| 3  | Life of Pie                 | 4           | 0       | 3  | Appleson     |
| 3  | Life of Pie                 | 4           | 0       | 4  | Squirrel     |
| 3  | Life of Pie                 | 4           | 0       | 5  | Yellow Flame |
| 4  | A Brief History Of Primates | 3           | 0       | 1  | NPH          |
| 4  | A Brief History Of Primates | 3           | 0       | 2  | Unpop        |
| 4  | A Brief History Of Primates | 3           | 0       | 3  | Appleson     |
| 4  | A Brief History Of Primates | 3           | 0       | 4  | Squirrel     |
| 4  | A Brief History Of Primates | 3           | 0       | 5  | Yellow Flame |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 1  | NPH          |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 2  | Unpop        |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 3  | Appleson     |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 4  | Squirrel     |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 5  | Yellow Flame |
| 6  | The Catcher in the Eye      | 1           | 1       | 1  | NPH          |
| 6  | The Catcher in the Eye      | 1           | 1       | 2  | Unpop        |
| 6  | The Catcher in the Eye      | 1           | 1       | 3  | Appleson     |
| 6  | The Catcher in the Eye      | 1           | 1       | 4  | Squirrel     |
| 6  | The Catcher in the Eye      | 1           | 1       | 5  | Yellow Flame |

|     |              |                 |      |   |   |              |
|-----|--------------|-----------------|------|---|---|--------------|
| 123 | H2 Computing | Ten Year Series | NULL | 0 | 1 | NPH          |
| 123 | H2 Computing | Ten Year Series | NULL | 0 | 2 | Unpop        |
| 123 | H2 Computing | Ten Year Series | NULL | 0 | 3 | Appleson     |
| 123 | H2 Computing | Ten Year Series | NULL | 0 | 4 | Squirrel     |
| 123 | H2 Computing | Ten Year Series | NULL | 0 | 5 | Yellow Flame |

Notice that there are two columns of ID. The first ID column is from Book table, while the second ID column is from the Publisher table.

**Think:**

*How can the attribute names in Book and Publisher be renamed to remove the confusion over the ID columns?*

The table above is a big table, with a lot of necessary rows. It is more useful to have the publisher of the title of each book. To do that, we can add a condition.

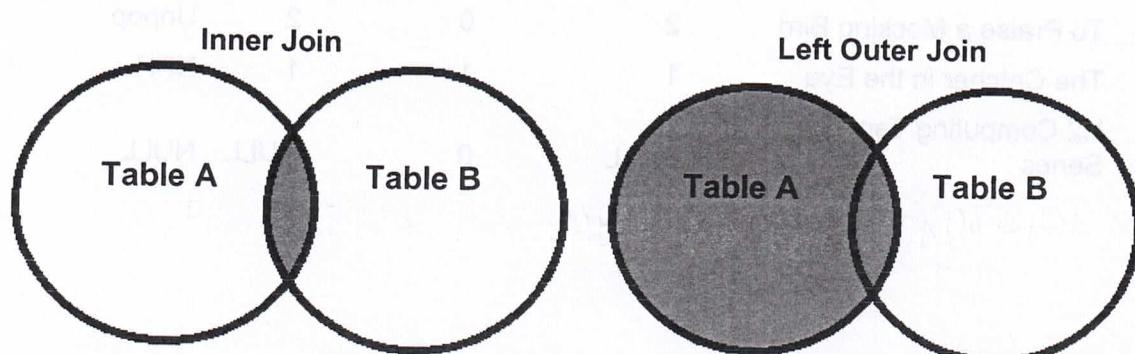
```
SELECT * FROM Book, Publisher
WHERE Book.PublisherID = Publisher.ID
```

The SQL statement above gives the following results:

| ID | Title                       | PublisherID | Damaged | ID | Name         |
|----|-----------------------------|-------------|---------|----|--------------|
| 1  | The Lone Gatsby             | 5           | 0       | 5  | Yellow Flame |
| 2  | A Winter's Slumber          | 4           | 1       | 4  | Squirrel     |
| 3  | Life of Pie                 | 4           | 0       | 4  | Squirrel     |
| 4  | A Brief History Of Primates | 3           | 0       | 3  | Appleson     |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 2  | Unpop        |
| 6  | The Catcher in the Eye      | 1           | 1       | 1  | NPH          |

This table is more meaningful as it links the titles to the publishers. However, notice that the title 'H2 Computing Ten Year Series' is not included as it has no PublisherID.

This is an example of inner join. Inner join can also be explicitly defined, like left outer join, which we will learn next.



Inner join selects all records from Table A and Table B which meet the join condition.  
 Left outer join selects all records from Table A, along with records from Table B which meet the join condition.

```
SELECT * FROM Book
INNER JOIN Publisher
ON Book.PublisherID = Publisher.ID
```

For example, the SQL statement above gives the following results:

| ID | Title                       | PublisherID | Damaged | ID | Name         |
|----|-----------------------------|-------------|---------|----|--------------|
| 1  | The Lone Gatsby             | 5           | 0       | 5  | Yellow Flame |
| 2  | A Winter's Slumber          | 4           | 1       | 4  | Squirrel     |
| 3  | Life of Pie                 | 4           | 0       | 4  | Squirrel     |
| 4  | A Brief History Of Primates | 3           | 0       | 3  | Appleson     |
| 5  | To Praise a Mocking Bird    | 2           | 0       | 2  | Unpop        |
| 6  | The Catcher in the Eye      | 1           | 1       | 1  | NPH          |

The results include all books who have publishers.

The title 'H2 Computing Ten Year Series' is not included as it has no publisher.

Notice that there are two columns of ID. The first ID column is from Book table, while the second ID column is from the Publisher table.

However, if the inner join is changed to left outer join as below:

```
SELECT * FROM Book
LEFT OUTER JOIN Publisher
ON Book.PublisherID = Publisher.ID
```

The results are different. This is because all records in Book are included in the left outer join, even books who have no publishers, unlike the inner join.

| ID  | Title                        | PublisherID | Damaged | ID   | Name         |
|-----|------------------------------|-------------|---------|------|--------------|
| 1   | The Lone Gatsby              | 5           | 0       | 5    | Yellow Flame |
| 2   | A Winter's Slumber           | 4           | 1       | 4    | Squirrel     |
| 3   | Life of Pie                  | 4           | 0       | 4    | Squirrel     |
| 4   | A Brief History Of Primates  | 3           | 0       | 3    | Appleson     |
| 5   | To Praise a Mocking Bird     | 2           | 0       | 2    | Unpop        |
| 6   | The Catcher in the Eye       | 1           | 1       | 1    | NPH          |
| 123 | H2 Computing Ten Year Series | NULL        | 0       | NULL | NULL         |

**Quiz**

1. Write down the SQL statement to join all the books that are not damaged with their Publishers.

*SELECT Book.Title AS "Book Title", Publisher.Name AS "Publisher Name"*

*FROM Book, Publisher WHERE Book.Damaged = 0 AND Book.PublisherID =*  
*Publisher.ID .*

---

**UPDATE**

The UPDATE command allows the editing of data values in a database. One or more records may be updated at the same time.

For example, the statement

`UPDATE Book Set Title = 'Book: ' || Title`

will update the values of Title in the Book table such that each book title now start with 'Book: '. Note the use of || for string concatenation (adding of two strings).

Consider another example. All the books borrowed by Kumara (BorrowerID 3) should have been borrowed by Sarah (BorrowerID 2). Then, the following SQL statement can be carried out to update the Loan table.

`UPDATE Loan Set BorrowerID = 2 WHERE BorrowerID = 3`

**Quiz**

1. All the books borrowed by Sarah (BorrowerID 2) should have been borrowed by Kumara (BorrowerID 3). Write down the SQL statement to update the Loan table.

*UPDATE Loan Set BorrowerID = 2 WHERE BorrowerID = 3*

---

**INSERT INTO**

The INSERT INTO command is used to insert a new record in a table.

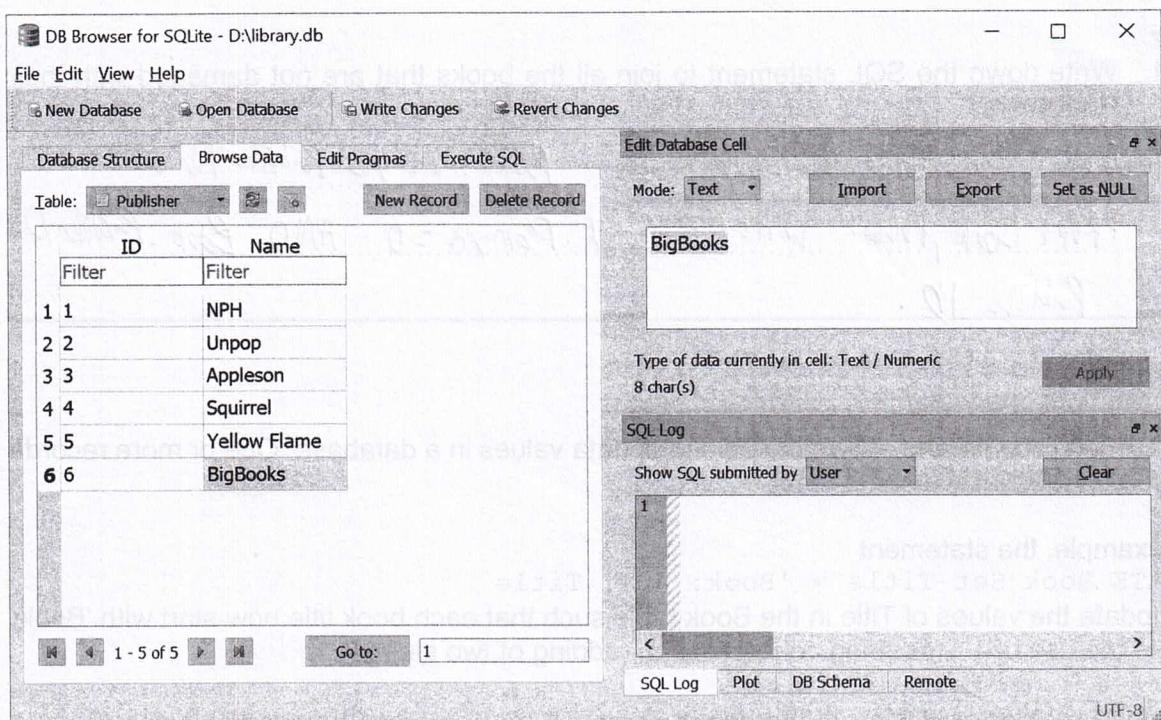
For example, to insert a new publisher, we can use the following SQL:

`INSERT INTO Publisher(Name) VALUES ('BigBooks')`

This will insert the publisher 'BigBooks' from the Publisher table.

You can check by selecting Publisher table under 'Browse Data' tab.

| ID  | Name      | Address                     | City          | State | Country | Phone        |
|-----|-----------|-----------------------------|---------------|-------|---------|--------------|
| 1   | Microsoft | One Microsoft Way           | Redmond       | WA    | USA     | 425-903-3600 |
| 2   | Oracle    | 500 Oracle Parkway          | Sunnyvale     | CA    | USA     | 408-526-2000 |
| 3   | IBM       | 1000 IBM Drive              | Chamblee      | GA    | USA     | 404-920-1000 |
| 4   | HP        | 3000 Hanover Street         | Palo Alto     | CA    | USA     | 415-852-1000 |
| 5   | SAP       | 1000 North Mathilda         | Burlingame    | CA    | USA     | 650-251-1000 |
| 6   | Siemens   | 1000 Siemens Drive          | Mountain View | CA    | USA     | 415-960-1000 |
| 7   | Intel     | 1000 West Esplanade         | Santa Clara   | CA    | USA     | 408-721-1000 |
| 8   | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 9   | AT&T      | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 10  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 11  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 12  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 13  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 14  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 15  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 16  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 17  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 18  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 19  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 20  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 21  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 22  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 23  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 24  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 25  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 26  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 27  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 28  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 29  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 30  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 31  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 32  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 33  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 34  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 35  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 36  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 37  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 38  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 39  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 40  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 41  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 42  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 43  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 44  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 45  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 46  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 47  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 48  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 49  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 50  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 51  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 52  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 53  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 54  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 55  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 56  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 57  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 58  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 59  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 60  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 61  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 62  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 63  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 64  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 65  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 66  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 67  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 68  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 69  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 70  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 71  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 72  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 73  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 74  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 75  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 76  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 77  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 78  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 79  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 80  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 81  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 82  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 83  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 84  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 85  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 86  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 87  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 88  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 89  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 90  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 91  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 92  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 93  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 94  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 95  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 96  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 97  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 98  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 99  | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |
| 100 | Verizon   | 1000 Avenue of the Americas | New York      | NY    | USA     | 212-847-1000 |



Notice that the ID is automatically given! Do you know how the database does it?  
(Hint: Answer is found later...)

## DELETE

The **DELETE** command is used to delete existing records in a table.

For example,

```
DELETE FROM Books WHERE Title = 'Life of Pie'
```

This will delete the book 'Life of Pie' from the Books table.

Please note that the text is case-sensitive, which means that the following 2 SQL statements are different.

```
DELETE FROM Book WHERE Title = 'Life of Pie'
```

```
DELETE FROM Book WHERE Title = 'Life of pie'.
```

Be careful when using the **DELETE** statement. For example:

```
DELETE FROM Loan
```

This will delete all entries from the Loans table!

## Quiz

- Which of the following SQL statements are valid? (Tick the statements)

|                                     |   |
|-------------------------------------|---|
| <input checked="" type="checkbox"/> | DELETE FROM Book WHERE Title = 'H2 Computing Ten Year Series' |
| <input type="checkbox"/>            | DELETE FROM Book WHERE Title = *                              |
| <input type="checkbox"/>            | DELETE FROM Book WHERE  |
| <input type="checkbox"/>            | DELETE FROM Book  |

## CREATE TABLE

The CREATE TABLE command allows you to create a table. You can see the SQL code for the various tables under Database Structure in DB Browser.

For example, to create the Borrower table, you can key in:

```
CREATE TABLE 'Borrower' (
    'ID' INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    'FirstName' TEXT NOT NULL,
    'Surname' TEXT NOT NULL,
    'Contact' INTEGER NOT NULL
)
```

Let's look at the SQL statement carefully.

INTEGER and TEXT are data types, 'ID', 'FirstName', 'Surname' and 'Contact' are field names. The ID field is the primary key of the table Borrower, while Autoincrement means the ID value is automatically given by the database. NOT NULL means that the fields do not accept Null values.

Thus, the syntax for creating tables in SQL is

```
CREATE TABLE table_name(
    column1_name COLUMN1_TYPE COLUMN1_CONSTRAINTS,
    column2_name COLUMN2_TYPE COLUMN2_CONSTRAINTS,
    ...
    PRIMARY KEY (column1_name, column2_name,...),
    FOREIGN KEY (column_name) REFERENCES table_name(column_name)
)
```

Also, note that there is an additional table sqlite\_sequence above.

If you view the table, you will see the following.

| name |           | seq |
|------|-----------|-----|
| 1    | Borrower  | 3   |
| 2    | Publisher | 6   |
| 3    | Loan      | 4   |

This table is used by SQLite so as to keep track of the next number to give for tables with AUTOINCREMENT. It is generated automatically when you have an AUTOINCREMENT field used in SQLite.

Let us look at the Book table next. The SQL code to create the table is:

```
CREATE TABLE 'Book' (
    'ID' INTEGER NOT NULL,
    'Title' TEXT NOT NULL,
    'PublisherID' INTEGER,
    'Damaged' INTEGER NOT NULL,
    FOREIGN KEY('PublisherID') REFERENCES 'Publisher'('ID'),
    PRIMARY KEY('ID')
)
```

Notice the line starting with FOREIGN KEY. It defines the foreign key, which is the field linking to the primary key of another table. For this example, it is linking the PublisherID field in the Book table to the ID field in the Publisher table.

### Quiz

5. Key in SQL statement to create a table Testing with fields name, tag\_no and remarks. The fields name and remarks should accept text, while tag\_no is an integer automatically incremented. The primary key is tag\_no. Name field should not be NULL.
- .....
- .....
- .....

### Tip!

You can click 'Revert Changes' in DB Browser to remove the recent changes made.

## DROP TABLE

The DROP TABLE command deletes the entire table.

For example, to remove the Loan table, you can key in:

```
DROP TABLE Loan
```

What is the difference between DELETE FROM Loan and DROP TABLE Loan?

With DELETE FROM Loan, you delete all entries from the Loan table, but the Loan table remains there. With DROP TABLE Loan, the Loan table will be removed. You cannot insert any entries into Loan table anymore.

### Quiz

6. Try keying DROP TABLE Publisher to remove the table containing the publishers. Is it possible? Why?
- .....
- .....

## Operators

You have seen some operators being used in the examples earlier. These operators are often used in SELECT statements, but can be used in other statements (like the UPDATE statement example shown earlier). The following are comparison operators, logical operators and arithmetic operators that you need to know.

### A. Comparison Operators

| Comparison Operator | Description   |
|---------------------|---|
| =                   | Checks if the values of two operands are equal or not, if yes then the condition becomes true.  |
| !=                  | Checks if the values of two operands are equal or not, if the values are not equal, then the condition becomes true.                        |
| <>                  | Checks if the values of two operands are equal or not, if the values are not equal, then the condition becomes true.                        |
| >                   | Checks if the values of the left operand is greater than the value of the right operand, if yes then the condition becomes true.            |
| <                   | Checks if the values of the left operand is less than the value of the right operand, if yes then the condition becomes true.               |
| >=                  | Checks if the value of the left operand is greater than or equal to the value of the right operand, if yes then the condition becomes true. |
| <=                  | Checks if the value of the left operand is less than or equal to the value of the right operand, if yes then the condition becomes true.    |

### B. Logical Operators

| Logical Operator | Description  |
|------------------|--|
| AND              | The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause. |
| OR               | The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.       |
| IS               | The value exists. For example, IS NULL looks for NULL values.                                    |
| IS NOT           | The value does not exist.  |
|                  | String concatenation   |

## C. Arithmetic Operators

| Arithmetic Operator | Name           | Description   |
|---------------------|----------------|---|
| +                   | Addition       | Adds values on either side of the operator  |
| -                   | Subtraction    | Subtracts the right-hand operand from the left-hand operand                       |
| *                   | Multiplication | Multiplies values on either side of the operator                                  |
| /                   | Division       | Divides the left-hand operand by the right-hand operand                           |
| %                   | Modulus        | Divides the left-hand operand by the right-hand operand and returns the remainder |

## Functions

Aggregate functions help to count / calculate results from the database.

| Function | Description       |
|----------|-------------------|
| MIN      | Minimum value     |
| MAX      | Maximum value     |
| SUM      | Sum of all values |
| COUNT    | Number of values  |

## Exercise

Open the file poly.db.

It contains information from Temasek Polytechnic Full-Time Enrolment Figures Breakdown, Annual dataset accessed on 23 March 2018 from Temasek Polytechnic which is made available under the terms of the Singapore Open Data Licence version 1.0 (<http://www.data.gov.sg>)

The database provides the enrolment in different polytechnic courses in Temasek Polytechnic (TP) by year and gender.

Try to give the SQL statement to generate the following.

1. List the courses in TP that has more than 250 males in 2017.

```
SELECT courl...nre FROM enrolmt WHERE c=50 AND  
gender = "Male" AND yr = 2017
```

2. List the courses in TP that has no more than 50 females in 2016.

```
SELECT courl...nre FROM enrolmt WHERE cont <=50  
AND gender > "Female" AND yr = 2016.
```

3. List out all fields in the table for courses not in School of Informatics & IT in 2016.

```
SELECT * from enrolmt WHERE school <> "School of  
Informatics & IT" AND year = 2016!
```

4. List the largest number of female(s) in a course in 2017.

```
SELECT MAX(cont) from enrolmt WHERE gender = "FEMALE"  
and yr = 2017
```

5. List the smallest number of male(s) in a course in 2016.

```
SELECT MIN(cont) FROM enrolmt WHERE gender = "MALE"  
AND year >= 2016 AND year <= 2017
```

6. Rename all the diplomas called 'Diploma in Cyber & Digital Security' to 'Diploma in Cyber and Digital Security'.

```
UPDATE diploma SET diploma = 'Diploma in Cyber and Digital  
Security' WHERE diploma = 'Diploma in Cyber & Digital Security'
```

7. Update the table to add 5 students to each gender and each course in School of Design in 2017.

UPDATE prnadm SET count = count + 5

WHERE year = 2017 AND school = 'School of Design'

## References

Notes are from MOE CPDD Computer Education Unit.

| Content | Links  |
|---------|--|
| SQL     | <a href="https://www.webucator.com/tutorial/learn-sql/simple-selects.cfm">https://www.webucator.com/tutorial/learn-sql/simple-selects.cfm</a><br><a href="https://www.tutorialspoint.com/sql/sql_and-or_clauses.htm">https://www.tutorialspoint.com/sql/sql_and-or_clauses.htm</a><br><a href="https://www.bbc.com/education/guides/z37tb9q/revision/5">https://www.bbc.com/education/guides/z37tb9q/revision/5</a><br><a href="https://www.w3schools.com/sql/default.asp">https://www.w3schools.com/sql/default.asp</a> |

## General Structure

SELECT \_\_\_\_\_

FROM \_\_\_\_\_

WHERE \_\_\_\_\_

GROUP BY \_\_\_\_\_

ORDER BY \_\_\_\_\_ ASC/DESC