# ISS Lab Tutorial: SQL

April 25,2023

## 1 What is SQL?

SQL (Structured Query Language) is a programming language used for managing and manipulating data in relational databases. A relational database stores information in tabular form, with rows and columns representing different data attributes and the various relationships between the data values. In this tutorial, we will introduce you to SQL queries, which are used to retrieve and update data in databases.

## 2 Initialisation

```
import sqlite3

class DBclass:
    def __init__(self, path):
        self.path = path
        self.db = sqlite3.connect(self.path)
        self.cur = self.db.cursor()

    def execute(self, query):
        self.cur.execute(query)
        return [i[0] for i in self.cur.description], self.cur.fetchall()
db = DBclass('ipl.db')
```

## 3 Some important SQL queries

### 3.1 Retrieving data from columns

This is a very basic query to display all data from a table. In this example, we are retrieving data from the table animal.

```
SELECT * FROM animal;
```

If you would like to only retrieve data from certain columns, list them after SELECT. In this example, we are retrieving data from the id and name columns.

```
SELECT id, name FROM animal;
```

### 3.2 Filtering data using WHERE clause

In addition to retrieving data from certain columns, you can also filter data by listing conditions after WHERE. In this example, there is one condition: age$\geq$2. We are looking for records with a value of 2 or more in the column age.

```
SELECT id, name, age FROM animal WHERE age>=2;
```

### 3.3 Filtering Data Using Conditions Joined by AND or OR Operator

If only one of the conditions needs to be met, you can use OR. If you want to filter data using more than one condition, you can use AND. In this example, we are looking for records with a value of 2 or more in the column age and 'dog' in the column name.

```
SELECT id, name, age FROM animal WHERE age >= 2 AND name = 'dog';
```

### 3.4 Using DISTINCT to Retrieve Non-Repeated Records

In this example, we want to retrieve records from columns name and color. If the values from these columns are the same in more than one record, the query returns only one of those records.

```
SELECT DISTINCT name, color FROM clothing;
```

### 3.5 Sorting Data According to Columns

To sort data according to a column, place the column name after ORDER BY. The default sorting method is alphabetical, but you can also display rows in descending order by adding DESC after the name of the column.

You can also sort data according to more than one column.

```
SELECT id, name FROM animal ORDER BY name DESC, id;
```

### 3.6 Using Mathematical Operators

You can write queries to calculate values by using mathematical operators like "+," "-," "*," and "/."

```
SELECT price - discount FROM product;
```

### 3.7 Combining data from different tables

You can join records from different tables using the operator UNION ALL. Remember that the records must be the same data type. In this example, we want to retrieve all rows with last names from the table customer and all rows with last names from the table employee.

```
SELECT last_name FROM customer UNION ALL SELECT last_name FROM employee;
```

INTERSECT returns the intersection of two sets of data. In this example, we only want to retrieve the last names listed in both tables.

```
SELECT last_name FROM customer INTERSECT SELECT last_name FROM employee;
```

### 3.8 Counting the Number of Rows in a Table

COUNT counts the number of rows. In this example, it returns the number of values from the column id stored in the table product (the number of all products).

```
SELECT COUNT(id) FROM product;
```

### 3.9 Calculating the Average of the Values in a Column

You can calculate the average of the values in a column using AVG. In this example, the query returns the average price of all products in the table product.

```
SELECT AVG(price) FROM product;
```

### 3.10 Calculating the Sum of the Values in a Column

SUM calculates the sum of the values in a column. In this example, it returns the value of all of the products.

```
SELECT SUM(price) FROM product;
```

## 3.11 Finding the Minimum and Maximum Value in a Column

You can find the minimum and maximum value stored in a column using MIN and MAX. In this example, the query returns the minimum price among the products.

```
SELECT MIN(price) FROM product;
```

## 3.12 Removing Data From a Table

DELETE FROM removes all data from a table. In this example, we want to delete all data from the table product.

```
DELETE FROM product;

DELETE FROM product WHERE id = 5;
```

## 3.13 Inserting Data Into a Table

You can add a new record to a table using INSERT INTO. After INSERT INTO, put the name of the table and then in brackets the names of the columns of the table. After that, put VALUES and then in the brackets the values for the columns. In this example, we want to insert 25 into id, 'sofa' into name, and 'furniture' into category in the table product.

```
INSERT INTO product(id, name, category) VALUES(25, 'sofa', 'furniture');
```

## 3.14 Updating a Column in a Table

UPDATE allows you to modify data in the records. After UPDATE, put the name of the table, then SET, and then the name of the column to modify with "=" and new value to insert. This query modifies all values in the column. In this example, we want to change all values in the column company to 'ABC'.

```
UPDATE product SET company = 'ABC';
```