Structured Query Language

# Data Definition Language (DDL)

# DROP TABLE

Remove tables if they already exist

DROP TABLE IF EXISTS books;

DROP TABLE IF EXISTS authors;

# CREATE TABLE

Create a table called authors with the following fields: author\_ID, which is the primary key and has an integer data type, firstname, surname and agent that have text data types

CREATE TABLE authors(

author\_ID INTEGER PRIMARY KEY,

firstname text,

surname text

agent text);

Create a table called books which links to the authors table using the author\_ID as the foreign key

CREATE TABLE books(

book\_ID INTEGER PRIMARY KEY,

author\_ID INTEGER,

title text,

publisher text,

year INTEGER,

genre text,

FOREIGN KEY (author\_ID) REFERENCES authors(author\_ID));

Chart

Description automatically generated with medium confidence

# Data Manipulation Language (DML)

We will use this book table in the examples that follow.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| BookID | Title | Author | YearPublished | Publisher | Genre |
| 1 | Fantastic Beasts .. | JK Rowling | 2001 | Bloomsbury | Fantasy |
| 2 | ..Chamber of Secrets | JK Rowling | 1998 | Bloomsbury | Fantasy |
| 3 | .. Order of the Phoenix | JK Rowling | 2003 | Bloomsbury | Fantasy |
| 4 | The BFG | Roald Dahl | 1982 | Penguin | Fantasy |
| 5 | Going Solo | Roald Dahl | 1986 | Jonathan Cape | Autobiography |
| 6 | Danny Champion .. | Roald Dahl | 1975 | Jonathan Cape | Children |
| 7 | War Horse | Michael Morpurgo | 1982 | Kaye & Ward | Historical fiction |
| 8 | Private Peaceful | Michael Morpurgo | 2003 | HarperCollins | Historical fiction |

# Select - To retrieve data from the table

To retrieve all records data from the table we can use the SELECT statement with the wild card operator \*.

SELECT \*

FROM tableName

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | Fantastic Beasts .. | JK Rowling | 2001 | Bloomsbury | Fantasy |
| 2 | ..Chamber of Secrets | JK Rowling | 1998 | Bloomsbury | Fantasy |
| 3 | .. Order of the Phoenix | JK Rowling | 2003 | Bloomsbury | Fantasy |
| 4 | The BFG | Roald Dahl | 1982 | Penguin | Fantasy |
| 5 | Going Solo | Roald Dahl | 1986 | Jonathan Cape | Autobiography |
| 6 | Danny Champion .. | Roald Dahl | 1975 | Jonathan Cape | Children |
| 7 | War Horse | Michael Morpurgo | 1982 | Kaye & Ward | Historical fiction |
| 8 | Private Peaceful | Michael Morpurgo | 2003 | HarperCollins | Historical fiction |

### Example

SELECT \*

FROM book

#### Retrieved data

We can also choose the fields that we wish to retrieve:

SELECT field1, field2, …

FROM tableName

|  |  |
| --- | --- |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| Harry Potter and the Chamber of Secrets | JK Rowling |
| Harry Potter and Order of the Phoenix | JK Rowling |
| The BFG | Roald Dahl |
| Going Solo | Roald Dahl |
| Danny Champion of the World | Roald Dahl |
| War Horse | Michael Morpurgo |
| Private Peaceful | Michael Morpurgo |

### Example

SELECT Author, Title

FROM book

#### 

We can sort the output of our SELECT statement by using the ORDER BY clause. ASC and DESC refer to sorting ascending and descending alphabetically or numerically of a specified field.

ORDER BY fieldname ASC|DESC

|  |  |
| --- | --- |
| Danny Champion of the World | Roald Dahl |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| Going Solo | Roald Dahl |
| Harry Potter and the Chamber of Secrets | JK Rowling |
| Harry Potter and Order of the Phoenix | JK Rowling |
| Private Peaceful | Michael Morpurgo |
| The BFG | Roald Dahl |
| War Horse | Michael Morpurgo |

### Example Sort ascending

SELECT Author, Title

FROM book

ORDER BY Title ASC

|  |  |
| --- | --- |
| War Horse | Michael Morpurgo |
| The BFG | Roald Dahl |
| Private Peaceful | Michael Morpurgo |
| Harry Potter and Order of the Phoenix | JK Rowling |
| Harry Potter and the Chamber of Secrets | JK Rowling |
| Going Solo | Roald Dahl |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| Danny Champion of the World | Roald Dahl |

### Example Sort Descending

SELECT Author, Title

FROM book

ORDER BY Title DESC

### DISTINCT

|  |
| --- |
| Michael Morpurgo |
| Roald Dahl |
| JK Rowling |

We can avoid selecting repeating data and can select distinct data using:

SELECT DISTINCT author FROM books;

### WHERE Clause

We can filter our selection using the WHERE clause

WHERE fieldname operator value

|  |  |
| --- | --- |
| **Operator** | **Description** |
| = | Value equal to |
| != | Value not equal to |
| < | Value less than |
| > | Value greater than |
| <= | Value less than or equal to |
| >= | Value greater than or equal to |

### SELECT using WHERE clause

### EXAMPLE 1 – Select books written since 2000

|  |  |  |
| --- | --- | --- |
| Fantastic Beasts … | JK Rowling | 2001 |
| Harry Potter … | JK Rowling | 2003 |
| Private Peaceful | Michael Morpurgo | 2003 |

SELECT Title, Author, yearPublished

FROM book

WHERE YearPublished > 2000

EXAMPLE 2 – Select books written by Michael Morpurgo

|  |  |
| --- | --- |
| War Horse | Michael Morpurgo |
| Private Peaceful | Michael Morpurgo |

SELECT Title, Author

FROM book

WHERE Author = “Michael Morpurgo”

Notice how the author name is in speech marks because it is a string datatype.

### Example 3 – Select by date

WHERE Date < DATE(“2010-01-01”)

### Boolean Operators

We can use Boolean and relational operators with the WHERE clause if we have multiple conditions that need to be met.

|  |  |
| --- | --- |
| **Operator** | **Description** |
| OR | Allows us to combine multiple conditions. Any of the conditions can be true for the overall expression to return true |
| AND | Allows us to combine multiple conditions. All conditions need to be true for the overall expression to return true |
| NOT | Reverses the value of a condition. If it is true it will be false and vice versa |

### EXAMPLE – Select all books written by Michael Morpurgo since 2016

|  |  |
| --- | --- |
| Private Peaceful | Michael Morpurgo |

SELECT Title, Author FROM book

WHERE Author=“Michael Morpurgo”

AND YearPublished > 2000

### BETWEEN Operator

We can use the between operator with eth WHERE clause to specify a range of values we wish to select

SELECT title, author from books WHERE Year BETWEEN 1986 AND 2001

This statement is equivalent to:

SELECT title, author from books WHERE Year >= 1986 AND Year <= 2001

|  |  |
| --- | --- |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| Going Solo | Roald Dahl |

### IN Operator

The IN operator allows us to select several values

SELECT title, authors FROM authors

WHERE author IN (“Michael Morpurgo”, “JK Rowling”)

It is the equivalent of:

SELECT title, authors FROM authors

WHERE author=“Michael Morpurgo” OR author=JK Rowling”

|  |  |
| --- | --- |
| War Horse | Michael Morpurgo |
| Private Peaceful | Michael Morpurgo |
| Harry Potter and the Order of the Phoenix | JK Rowling |
| Fantastic Beasts and Where to Find Them | JK Rowling |

### LIKE Operator

The LIKE operator allows us to select a common pattern and can be use in conjunction with the wild card %.

SELECT title, authors FROM authors

WHERE author LIKE “%i%”;

This statement selects all records that has the letter “i” in the author name.

|  |  |
| --- | --- |
| Harry Potter and the Order of the Phoenix | JK Rowling |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| War Horse | Michael Morpurgo |
| Private Peaceful | Michael Morpurgo |

# Update - To update records in a database

To make changes to a record that is already in a table we can use the UPDATE statement.

EXAMPLE 1: Update the book table to change the genre of all fields to Children

UPDATE book

SET Genre=“Children”

EXAMPLE 2: Update the book table to change the author name from JK Rowling to Joanne Rowling.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Book ID** | **Title** | **Author** | **Year Published** | **Publisher** | **Genre** |
| 1 | Fantastic Beasts . | Joanne Rowling | 2001 | Bloomsbury | Children |
| 2 | Harry Potter .. | Joanne Rowling | 1998 | Bloomsbury | Children |
| 3 | Harry Potter .. | Joanne Rowling | 2003 | Bloomsbury | Children |
| 4 | The BFG | Roald Dahl | 1982 | Penguin | Children |
| 5 | Going Solo | Roald Dahl | 1986 | Jonathan Cape | Children |
| 6 | Danny . | Roald Dahl | 1975 | Jonathan Cape | Children |
| 7 | War Horse | Michael Morpurgo | 1982 | Kaye & Ward | Children |
| 8 | Private Peaceful | Michael Morpurgo | 2003 | HarperCollins | Children |

UPDATE book

SET Author=“Joanne Rowling”

WHERE Author=“JK Rowling”

# INSERT INTO - Adding new records

INSERT INTO is a commonly used command in SQL for adding new records to database tables. To insert all attributes for a table we can use:

INSERT INTO table

VALUES(value1, value2,…)

### Example

INSERT INTO book

VALUES (‘Boy’, ‘Roald Dahl’, 1984, ‘Penguin’, ‘Autobiography’)

Sometimes we do not enter data into every field. Instead we can explicitly state which fields we would like to add the data to.

INSERT INTO table (field1, field2,…)

VALUES(value1, value2,…)

The values correspond to the fields in the table i.e.:

* Field 1: Book ID
* Field 2: Title
* Field 3: Author
* Field 4: YearPublished
* Field 5: Publisher
* Field 6: Genre

### Example

INSERT INTO book (Title, Author, YearPublished, Publisher, Genre)

VALUES (‘Boy’, ‘Roald Dahl’, 1984, ‘Penguin’, ‘Autobiography’)

# Deleting records

To delete a record we specify which record(s) from which table we wish to remove.

DELETE FROM table WHERE condition

### Examples

Remove all books

DELETE FROM book

The WHERE clause is used to filter records so that we do not apply a statement to a whole table.

Remove all books written by JK Rowling:

DELETE FROM book WHERE Author=‘JK Rowling’

Remove all books written by Michael Morpurgo and written before 2000

DELETE FROM book WHERE Author=‘Michael Morpurgo’ AND YearPublished < 2000

# Select Attributes from multiple tables

We will use the following database table as an example case study.

Table

Description automatically generated

We need to specify that we only wish to select the records where the primary key and foreign key match.

Examples

|  |  |
| --- | --- |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| Harry Potter and the Chamber of Secrets | JK Rowling |
| Harry Potter and Order of the Phoenix | JK Rowling |
| The BFG | Roald Dahl |
| Going Solo | Roald Dahl |
| Danny Champion of the World | Roald Dahl |
| War Horse | Michael Morpurgo |
| Private Peaceful | Michael Morpurgo |

Retrieve data book title and author surname

SELECT book.Title, author.Surname

FROM author, book

WHERE author.AuthorID=book.AuthorID

Retrieve book title and author surname where genre is *fantasy*

|  |  |
| --- | --- |
| Fantastic Beasts and Where to Find Them | JK Rowling |
| Harry Potter and the Chamber of Secrets | JK Rowling |
| Harry Potter and Order of the Phoenix | JK Rowling |
| The BFG | Roald Dahl |

SELECT book.title, author.surname

FROM author, book

WHERE author.AuthorID=book.AuthorID

AND book.Genre=“Fantasy”

Retrieve book title and author surname where genre is fantasy and sort in descending order Title

SELECT book.title, author.surname

|  |  |
| --- | --- |
| The BFG | Roald Dahl |
| Harry Potter and Order of the Phoenix | JK Rowling |
| Harry Potter and the Chamber of Secrets | JK Rowling |
| Fantastic Beasts and Where to Find Them | JK Rowling |

FROM author, book

WHERE author.AuthorID=book.AuthorID

AND book.Genre=“Fantasy”

ORDER BY title DESC

### INNER JOIN

We can also use the INNER JOIN clause to select data from a pair of tables that have the same values

SELECT books.title, authors.name from authors

INNER JOIN books ON authors.author\_ID=books.author\_ID;

This also produces the same result

SELECT books.title, authors.name from authors, books

WHERE authors.author\_ID=books.author\_ID;