SQL EXERCISES

EXERCICE 1

Part 1 :

* Get all the information from each books

SELECT \* FROM books

* Get the title of each book

SELECT title FROM books

* Get the title and publication date of each book

SELECT title, publication\_date FROM books

Part 2 :

* Retrieve the title of each book, sorted alphabetically

SELECT title FROM books ORDER BY title asc

* List all authors in alphabetical order and without duplicate

SELECT Distinct name FROM authors ORDER BY name asc

Part 3:

* Recover the book with id 5

SELECT \* FROM books WHERE id= 5

* Find books released between 2000 and 2010

SELECT \* FROM books WHERE publication\_date BETWEEN "2000-01-01" AND "2010-12-31";

* Find books that did not come out between '2000' and '2010'

SELECT \* FROM books WHERE publication\_date NOT BETWEEN "2000-01-01" AND "2010-12-31";

Part 4:

* Recover all books starting with "Lord of the ring"

SELECT \* FROM books WHERE title LIKE 'Lord of the ring%';

* Get all the books from the author "Tolkien" (you have to know the author id)

SELECT \* FROM books WHERE author\_id = 2

* Get all the author's books that are not "Tolkien"

SELECT \* FROM books WHERE author\_id != 2

Part 5:

* Recover all books from newest to oldest

SELECT \* FROM books ORDER BY publication\_date DESC

* Count the number of books by each author

SELECT author\_id, COUNT(\*) FROM books GROUP BY author\_id

Part 6

* List only the first 5 books

SELECT \* FROM books ORDER BY publication\_date LIMIT 5;

* List the last 4 books (from newest to oldest)

SELECT \* FROM books ORDER BY publication\_date DESC LIMIT 4;

Part 7:

* Get the sum of all the book's price (total amount of all books)

SELECT SUM(price) AS total FROM books

* Get the sum of books price PER author\_id.

SELECT author\_id, SUM(price) AS total FROM books GROUP BY author\_id

Part 8:

* Update all books with author\_id = 2. Those books all cost now 10.

UPDATE books SET price = 10 WHERE author\_id = 2

* Update all books with author\_id = 5. Those books all cost now 5 more euro than previous price.

UPDATE books SET price = price + 5 WHERE author\_id = 5

EXERCICE 2

Now, there is a 'novels' table that has the same fields as the 'books' table.

- Retrieve all novels and all books in one query.

SELECT \* FROM books UNION SELECT \* FROM novels;

- Collect all novels and all books from/since 2018

SELECT \* FROM books WHERE publication\_date >= '2018-01-01' UNION SELECT \* FROM novels WHERE publication\_date >= '2018-01-01';

- Get the total price per author (books and novels combined).

SELECT table1.author\_id, SUM(table1.price) FROM (SELECT author\_id, price FROM books UNION ALL SELECT author\_id, price FROM novels) table GROUP BY table1.author\_id

- Get the total price for all books AND all novels (books and novels seperated).

SELECT 'books' as Origin, SUM(price) as Total FROM books UNION SELECT 'novels' as Origin, SUM(price) as Total FROM novels

SQL JOIN

EXERCICE 1

Part 1:

- For each book, Retrieve its title and the name of the author

SELECT title, name FROM books INNER JOIN authors ON books.author\_id = authors.id

- Get the title of each book and the name of the author, sorted alphabetically on the title of the book.

SELECT title, name FROM books INNER JOIN authors ON books.author\_id = authors.id ORDER BY title

- Same but sorted on the author's name.

SELECT title, name FROM books INNER JOIN authors ON books.author\_id = authors.id ORDER BY name

- For each book, get all the information about it and the name of the author

SELECT books.\*, authors.name FROM books INNER JOIN authors ON books.author\_id = authors.id

Part 2:

- Retrieve the category and the name of the song for each song, but we still want to see all the categories, even if they don't have one or more matching songs.

SELECT type, title FROM categories LEFT JOIN songs ON categories.id = songs.categ\_id

- Retrieve the name of the artist, the song name and the category for each song.

SELECT name, title, type FROM artists INNER JOIN songs ON artists.id = songs.artist\_id INNER JOIN categories ON songs.categ\_id = categories.id

- Retrieve the name of the artist, the song name and the category for each song, but we still want to see all the categories without matched songs.

SELECT name, title, type FROM artists INNER JOIN songs ON artists.id = songs.artist\_id RIGHT JOIN categories ON songs.categ\_id = categories.id

EXERCICE 2

Part 1 :

- Retrieve books that are under 50 years old (released less than 50 years ago).

SELECT \* FROM books WHERE YEAR(NOW()) - YEAR(publication\_date) < 50

- Retrieve all the authors that are Female (gender)

SELECT \* FROM authors WHERE gender = "Female"

Part 2:

- Get all the books from the author "JK Rowling"

SELECT title FROM books b INNER JOIN authors a ON b.author\_id = a.id WHERE a.name LIKE '%J.K. Rowling%'

- Get all the author's books that are not "JK Rowling"

SELECT title FROM books b INNER JOIN authors a ON b.author\_id = a.id WHERE a.name NOT LIKE '%J.K. Rowling%'

- Retrieve all books whose author has the letter 's' in his name or first name.

SELECT title FROM books b INNER JOIN authors a ON b.author\_id = a.id WHERE a.name LIKE '%s%'

Part 3:

- Retrieve the cheapest book with author's name

SELECT title, name FROM books b INNER JOIN authors a ON b.author\_id = a.id ORDER BY price LIMIT 1;

- Retrieve the most expensive book with author's name

SELECT title, name FROM books b INNER JOIN authors a ON b.author\_id = a.idORDER BY price DESC LIMIT 1;

- Retrieve the 3 most expansive books with author's name

SELECT title, name FROM books b INNER JOIN authors a ON b.author\_id = a.id ORDER BY price DESC LIMIT 3;

Part 4:

- Get all the books from the youngest to the oldest author

SELECT title, name FROM books b INNER JOIN authors a ON b.author\_id = a.id ORDER BY birth\_year DESC

- Retrieve the number of books by author (with the name of the author)

SELECT name, COUNT(\*) FROM books b INNER JOIN authors a ON b.author\_id = a.id GROUP BY a.id

Part 6:

- Get the sum of price for all the books from the author 'JK Rowling'

SELECT SUM(price) FROM books b INNER JOIN authors a ON b.author\_id = a.id WHERE a.name LIKE '%J.K. Rowling%'

- Get the sum of price for all books by author

SELECT name, SUM(price) FROM books b INNER JOIN authors a ON b.author\_id = a.id GROUP BY a.id

- Get average price of all books by author

SELECT name, AVG(price) FROM books b INNER JOIN authors a ON b.author\_id = a.id GROUP BY a.id

- Get the sum of the price of all books by gender

SELECT gender, SUM(price) FROM books b INNER JOIN authors a ON b.author\_id = a.id GROUP BY gender

Part 7:

- Retrieve gender that has more than 3 books

SELECT gender, COUNT(\*) as Total FROM books b INNER JOIN authors a ON b.author\_id = a.id GROUP BY gender HAVING Total > 3

Part 8:

- All JK Rowling's books have made 20 more price. Update these books.

UPDATE books b INNER JOIN authors a ON b.author\_id = a.id SET price = price + 20 WHERE a.name LIKE '%J.K. Rowling%'