Case Study – Combining Tables

Welcome to Books & Books! Now that we have a couple of tables in our database, we can start to create more data and perform more meaningful joins.

We now need to build invoices data from our stores. Books and Books will start to sell books at our Pagey and Library stores and we need to have our system ready to start selling!

**Hint:** To make this exercise not dependent on previous exercises, use the recreate\_books.sql script from the folder 5-Combining Tables / Case Study / recreate\_books.sql and make sure you run the code before attempting this case study. If you have completed the past case studies with success, you just have to continue working on that database.

[**🔥** – Exercises that may have not been covered throughout the lectures and may require some research/google searches!](https://emojipedia.org/fire/)

**Exercise 1 – Create a table named invoices with the following columns**

* **invoice\_id, a integer primary key.**
* **store\_id, a integer column that is not null**
* **BookId, a integer column that is not null**
* **A invoice\_date, a Date type column**
* **price, a decimal(19,4) type column.**

**Exercise 2 – Insert three invoices into the invoices table:**

* **invoice\_id 19994 on store\_id 2 where the customer purchased BookId 3. This book was purchased on the 1st of January of 2021 at a price of 19$.**
* **invoice\_id 19995 on store\_id 4 where the customer purchased BookId 7. This book was purchased on the 3rd of March of 2020 at a price of 1.5$.**
* **invoice\_id 19996 on store\_id 4 where the customer purchased BookId 15. This book was purchased on the 5th of April of 2021 at a price of 21.50$.**

**Exercise 3 – Perform the appropriate join to add the book name to the invoices table – only add the common products between the invoices and the books.**

**🔥Exercise 4 – On top of the join performed above, add the store name to the result. Hint: You can stack joins by immediately adding a new “inner join” clause right after the end of the last query with a new table with a new alias.**

**Exercise 5 – Perform the appropriate join to add the book name to the invoices table – retain all the books that were sold, even if they don’t have a name in the books table.**

**Exercise 6 – Create a table named invoices\_new with the same structure as the invoices table and a new column called discount which should be a decimal (19,4).**

**Exercise 7 – Insert two invoices into the invoices\_new table:**

* **invoice\_id 19994 on store\_id 2 where the customer purchased BookId 3. This book was purchased on the 1st of January of 2021 at a price of 19$ with a discount of 0$.**
* **invoice\_id 19997 on store\_id 2 where the customer purchased BookId 1. This book was purchased on the 3rd of November of 2020 at a price of 2$ and a discount of 1$.**

**Exercise 8 – Create a query result with the invoices from the invoices and invoices\_new tables in the same table. Retain all duplicate invoices and all columns common to both tables.**

**Exercise 9 - Create a query result with the invoices from the invoices and invoices\_new tables in the same table. Collapse all duplicate invoices into a single row and subset only the invoice\_id and price.**

**Exercise 10 – Add a column discount with type decimal(19,4) on the invoices table.**

**🔥Exercise 11 – On the result from exercise 8 and adding the discount column, create a new column in the resulting set with the discount divided by the price. Name that column price\_discount.**