Homework 4
DD1368
Rafael Bechara & Paula Hanna

1. Published_Book (Title: String, Edition: String, publisher: String, DateOfPublish: Date)

2- Book(PhysicalBook_ID: Integer, Title: String, Edition: String, ISBN: String, language: String)

3- Damaged_Book(PhysicalBook_ID: integer, Damage: string)

4- Prequel_And_Sequel(PhysicalBook_ID: Integer,Title : String , prequel : String , sequel: String)

5- Book_Author(PhysicalBook_ID: integer, Author: string)

6- Book_Genre(PhysicalBook_ID: integer, Genre: string)

7- Student(StudentID : Integer,FirstName: string, LastName: string , Email: string, Program: string, Address: string)

8- Admin(AdminID: integer, FirstName: string, LastName: string, Email: string, Adress: string, Department: string, PhoneNumber: string)

9- Borrowed_Book(Borrowing_ID: Integer, DateOfBorrowing: Date, ExpiringDate: Date, ReturnDate: Date, PysicalBook_ID: integer, Student_ID: integer)

10- Fines(Borrowing_ID: integer, FineAmount: double)

11- Transaction_Information(Transaction_ID: integer, DateOFPayment: Date, PaymentMethod: string, Borrowing_ID)

1) **π** PhysicalBook_ID(Borrowed_Book) **-** **π** PhysicalBook_ID (**σ** return_date <= current_date (Borrowed_Book))

To get a list of all books that are being currently borrowed we need to list all the books that have been borrowed at any time by projecting the PhysicalBook_ID attribute from the Borrowed_Book relation and we need also to list all the books that have been returned until today's date, we do that by projecting the PhysicalBook_ID attribute from the Borrowed_Book with the selection condition that the return date must be less or equal than the current date. taking the set difference between the first and the second list results in a list of all books that are being currently borrowed.

2) **π** Student_ID (Student) **-** **π** Student_ID (borrowed_ book)

To get a list of all users that have not yet borrowed a single book we need to take the set difference between the projection of the Student_ID attribute in the Student relation and the projection of the Student_ID in the borrowed_ book relation(gives a list of all student_ID that have at sometime borrowed a book)

Homework 4
DD1368
Rafael Bechara & Paula Hanna

OBS we did not include Admins in our relational algebra expression because our database has it so that only students can loan books which Dena has accepted. But in case of inclusion of admins we just need to make a union between the result of our relational algebra statement and the projection of the Admin_Id from the Admin relation.

3) **π** PhysicalBook_ID(Book) - **π** PhysicalBook_ID(Borrowed_Book)

To get a list of all book that have not been borrowed yet, we simply need to project the difference between Book.PhysicalBook_ID and Borrowed_Book.PhysicalBook_ID, i.e. all books that are in the Book table but not in the Borrowed_Book table.
The final result will be a list of PhysicalBook_ID for all books that have not been borrowed yet.

4) **π** Student_ID (**σ** No_Of_Fines >= 4 (ρUser_Fine_No (Student_ID , No_Of_Fines) Student_ID $\mathcal{F}_{\text{count FineAmount}}$ (**π** Student_ID , FineAmount (Borrowed_Book ⋈ Fines))))

To get a list of all users that have 4 or more fines we need first to natural join the Borrowed_Book relation and the Fines relation , the natural join occurs on the equality of the Borrowing_ID which is PK for Borrowed_Book and FK for Fines, then we project only Student_ID and FineAmount because that is all that we apply an aggregate function on that relation by grouping by student_ID and using the count function on FineAmount, what this does is create a list in which we keep track of how many time a certain user gets a fine, and we clarify that by using the rename operation, where we create a new relation User_Fine_No and change the name of the FineAmount attribute to No_Of_Fines attribute so it is easier to understand what values the new table store. After that we just project Student_ID from this new relation with the selection condition that the No_Of_Fines attribute is more than equal to 4.

5) **π** FirstName, LastName (**σ** Title = 'Harry Potter and the prisoner of azkaban' ^ return_date >= '2015-01-01' ^ return_date <= '2020-12-31' ((Borrowed_Book ⋈ Book) ⋈ Student))
To get the first and last name of the students who returned the third Harry Potter book (Harry Potter and the prisoner of azkaban) we needed to use natural join between Borrowed_Book and Book tables (based on the fact that both tables have PhysicalBook_ID) and then we needed to use natural join again between the result from the first natural join and the Student relation (based on the fact that the first natural join will give us a table which has Student_ID as an attribute). Now the final relation we get after those joins will have the following attributes (PhysicalBookID, Title, ISBN, Language, Borrowing_ID, DateOfBorrowing, ExpiringDate , ReturnDate , Student_ID, First_Name, Last_Name, E-mail, Program, Address). by applying the filter "Title = Harry Potter and the prisoner of azkaban" and "return_date >= '2015-01-01" and "return_date <= '2020-12-31'" and project First_Name and Las_Name we get a list of all students (users) that returned the third Harry Potter book between the year 2015 and 2020.

Homework 4
DD1368
Rafael Bechara & Paula Hanna

6) $\boldsymbol{\pi}$ Title ($\boldsymbol{\sigma}$ DateOfPublish < ‘2010-01-01’ ^ Genre = ‘Horror’ ((Book_Genre ⋈ Book) ⋈ Published_Book)) ∩
$\boldsymbol{\pi}$ Title ($\boldsymbol{\sigma}$ DateOfPublish < ‘2010-01-01’ ^ Genre = ‘Fantasy’ ((Book_Genre ⋈ Book) ⋈ Published_Book))

To get the title of all books that have been released before 2010 and have fantasy and horror genres we need first to natural join Book_Genre, Book and the join occurs on the equality of Physical_Book_ID and then natural join with Published_Book that occurs on the equality of Title and Edition , which gives all relevant attributes. Then we project Title from the new relation with the selection condition that DateOfPublish is less than 2010 and the genre is equal to ‘Horror’.
Now we redo this operation where we project Title from the new relation But the selection condition is now that DateOfPublish is less than 2010 and the genre is equal to ‘Fantasy’.
Using the intersection operation between these two expressions gives us the title of all books that have been released before 2010 and have fantasy and horror genres.