

Project Book On SQL Online Book Recommendation System



*Submitted to University College Dublin, for the subject
Introduction to Relational Databases and SQL*

(COMP40725)

Academic year 2018-19

Prof Tony Veale

-Sumit Chawan, MSc Computer Science.

Student No : 18200549

Table of Contents

| | |
|---|----|
| 1. Introduction | 1 |
| 2. Database Plan: A Schematic View | 3 |
| 3. Database structure: A Normalized View | 8 |
| 4. Database Views..... | 11 |
| 5. Procedural Elements | 15 |
| 6.Example Queries : Your database in action..... | 20 |
| 7.Conclusion..... | 23 |
| 8. Acknowledgements..... | 24 |
| References | 24 |
| | |
| Figure 1: E-R Diagram for ‘Online Book Recommender System’ | 4 |
| Figure 2: view for award winning books | 12 |
| Figure 3: View on the old- classic books | 13 |
| Figure 4: Output for query on top rated books for children | 14 |
| Figure 5: Output for query on top rated books for Males | 14 |
| Figure 6: Query to create a view for Popular sellers..... | 15 |
| Figure 7: output for top rated books | 15 |
| Figure 8: Gender Trigger..... | 16 |
| Figure 9: get_country_id function | 17 |
| Figure 10: Stored procedure for recommendation based on region | 19 |
| Figure 11: Stored procedure output for recommendation based on region | 19 |
| Figure 12: Stored procedure for personalized reader recommendations | 20 |
| Figure 13: Personalized user recommendations..... | 20 |
| Figure 14: Non-personalized : Top Recommended Books | 21 |
| Figure 15: Query for top recommender’s author | 21 |
| Figure 16: Books based on top themes..... | 22 |
| | |
| Table 1: Description of Database Attributes | 4 |

1. Introduction

In the past few years, there has been a tremendous increase in the data that's being generated from multiple sources like social media and e-commerce systems leading to the enormous data at disposal. This outburst in the amount of digital information has given rise to the problem of information overload and undoubtedly creates hindrance in the process to access relevant information in a timely manner. A number of information retrieval systems like google, yahoo, AltaVista have made the process a little lighter but they do not cater to the potential user requirements or personalized preferences. The incapability of these systems has led to the much and ever growing need of the Recommender systems. The Recommender systems helps to tackle the problem of information overload by filtering out the important information (Isinkaye, 2015) from large amount of data that is generated is generated at a fly as per the user preferences.

Recommender systems helps in making the predictions for a user ,by predicting whether a user would buy a item per se based on his buying habits, his profile on the e-commerce website, recent browsed products and even considering the association and influence of the people linked to him. The e-commerce system like amazon provide a platform for both the consumers and the retailers by providing consumers with a variety of products from which they can search from .At the same time it boasts the retailer sells by implementing customer-loyalty, cross-selling and identifying the target audience for the products by providing a proper set of recommendations. They help the audience a access to a variety of products they might not find offline or for that matter would be extremely difficult to find online even though they are available by providing precise and relevant recommendations.

The aim of the project is to build a database schema which would suffice the functional aspects of an online book recommender system. A structured database will help in organizing the data that needs to be accessed in order to provide recommendations to the user .

The project makes use of the relational MYSQL database system that will form the underlying schema to hold the data for an online Book Recommendation System. The data used for making recommendation is comparatively very less as that expected from an e-commerce system but can provide a basic infrastructure in a way to provide recommendations. The system does not aim at making the full -fledge functional recommender system as the domain is quite restricted in terms of computation or algorithms that are used to provide recommendations.

The recommender system will provide a basic set of functionality in order to make recommendations based on the available data in various tables . The system makes use of various views, stored procedures and queries that can help to generate recommendations which are non-personalized and personalized .Personalized recommendation are the once which are based entirely on the user preferences like the books purchased by the user in the past, the types of books that the user tends to read based on their qualities and the underlying themes like politics, education, romance etc. The non -personalized recommendations are the ones that are derived based on the similarity of a reader with his peers or generic qualities like the age group, gender the reader belongs to.

The user-based collaborative filtering can be thought as a best example of non-personalized recommendation which depends on the (Dr. Michael O'Mahony, no date) "word of mouth process". It is mainly dependent on the explicit form of preferences like the ratings provided by the user on a pre-determined scale of say - {1-5} or [2] gathered implicitly from the user behavior like sites visited , topics searched etc. The database therefore maintains the structure to store the search history of the user, the ratings provided by the user for the books read etc.

The database also aims to provide views that can be accessed for quick retrieval of relevant information in contrast to the traversal of entire table frequently to provide generic and prompt recommendations. The use of stored procedure and triggers is also made use of for simplification of dynamic and complex query processing and validations of certain columns prior to making inserts or updates to the tables.

2. Database Plan: A Schematic View

The database for the book recommender system consists of tables that are necessary to capture the user (reader) information and item information (books) for the purpose of making recommendations. The designing of the database is done by taking into consideration the major entities in the system which are the books and the user. The book entity is the prime entity which is needed to be recommended to the user entity. The books are represented by the table 'title' which is the central table to the database which contains the ISBNs- International standard Books Number and its book title. The ISBN is a 10/13 digit number which is universally accepted and is unique for each and every book that is published. The ISBN acts as a foreign key to almost all the entities in the schema by linking the titles table to the themes, qualities and authors table.

The other central entity to the database is the 'user'(reader), the entity to which the recommendations are to be made . The 'user' entity consists of user_id and other attributes like first_name , address which define the user.

On the initial set of tables author, themes, qualities and titles that were provided a few more essential tables like the user, user_rating, book_properties, purchase are constructed which are linked together with two keys the isbn and the user_id as per the application.

The database is spaced out keeping in mind of the potential alterations and future scope of the system. The focus in the schema design is on the use of independent entities by making use of various constraints for example : Instead of storing the city _name directly into the user table its id is stored, this in turn facilitates two important aspects 1) The space in storing the character string that's tends to repeat for multiple rows . 2) The amount of computation required on the updating even in the case of minor changes.

The ER- Diagram can be said to be a graphical representation of the (Souce:<https://searchdatamanagement.techtarget.com/definition/entity-relationship-diagram-ERD>) entities in the information schema, that also describe the relationships that exists amongst them. The ER-Diagram for the book

recommender system depicts the entities , their relationships, triggers and views that represent the system .

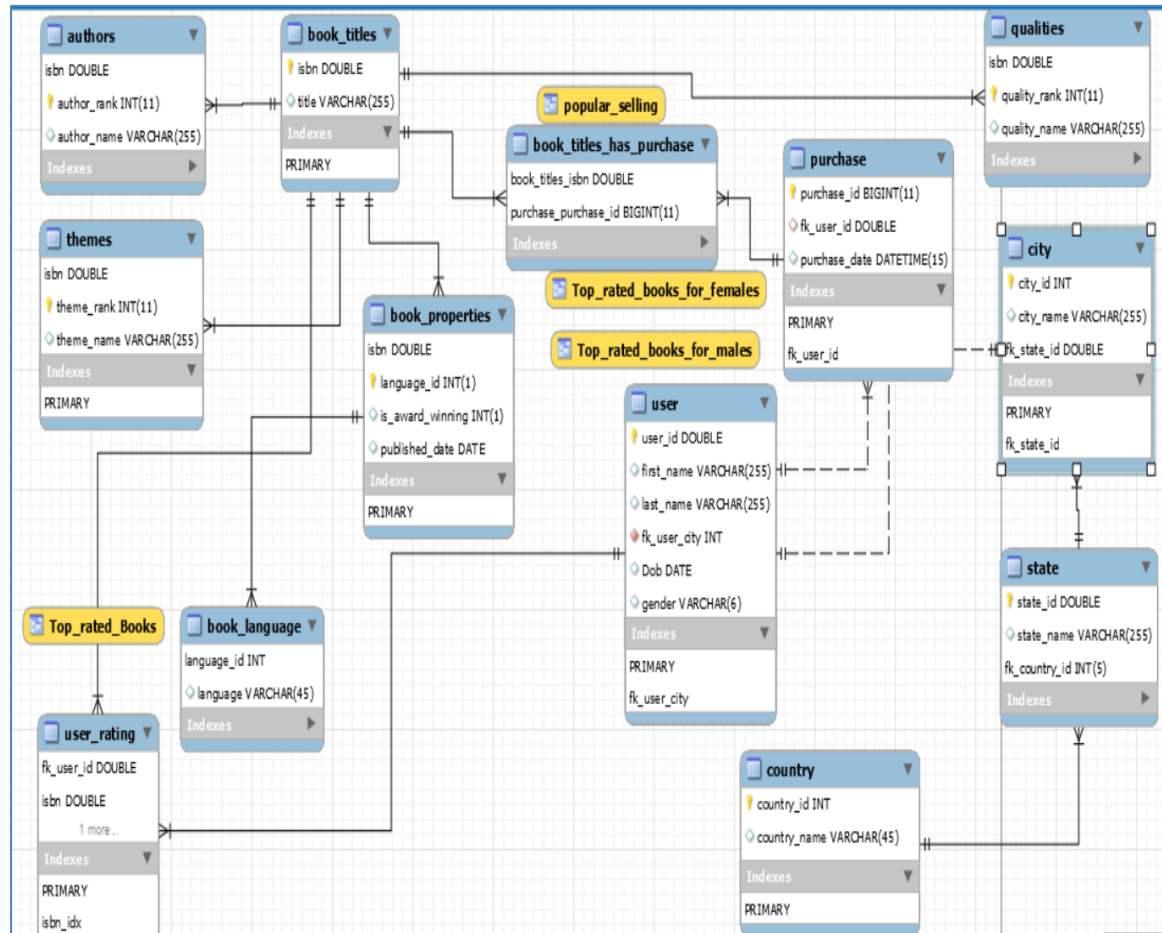


Figure 1: E-R Diagram for 'Online Book Recommender System'

The **views** represented in the **yellow** rectangular boxes are explained in the later part of the report that is the section 4. Given below is the description of the entities and its attribute in the recommender system schema.

Table 1: Description of Database Attributes

| Database Table | Description |
|----------------|--|
| book_titles | <p>This table is the central table in the schema as all the titles are what would be recommended to the end user .</p> <p>Attributes : 1) ISBN : The isbn acts as the primary key in the table and is taken to be double</p> |

| | |
|-----------|--|
| | <p>2) title : The title consists of the name of the books to be recommended or say present in the entire database.</p> |
| authors | <p>The author table is the one which contains the information of the writer of the books and is linked to the book_titles using isbn which in combination with author_rank act as the primary key to the table.</p> <p>Attributes : 1) isbn</p> <p>2)author_rank : A book can be authored by multiple authors and all the authors are given a rank for the authorship of the book based on their contribution toward the book.</p> <p>3)author_name : The person who has authored the book</p> <p><i>primary_key(isbn,author_rank)</i></p> |
| themes | <p>Theme table describes the actual theme on which the book is based.</p> <p>Attributes : 1) isbn</p> <p>2) theme_rank : A book can have multiple themes and hence have been given a ranks just like the author_rank column.</p> <p>3) theme_name : The name of the theme. like for example love, politcis,fantasy</p> <p><i>primary_key(isbn,theme_rank)</i></p> |
| qualities | <p>Qualities table describes the qualities of the book.</p> <p>Attributes : 1) isbn</p> <p>2) quality_rank : A book can have multiple qualities and hence have been given a ranks just like the author_rank column.</p> <p>3) quality_name : The name of the theme. like for</p> |

| | |
|-----------------|--|
| | <p>example love, entertaining, popular etc.</p> <p><i>primary_key(isbn,quality_rank)</i></p> |
| user | <p>The reader to whom the recommendations are to be made</p> <p>Attributes- 1)user_id :uniquely identifies each user in the system</p> <p>first_name & last_name</p> <p>Dob : The users date of birth which can be used to identify target audience as per the age groups to make recommendations.</p> <p>Gender : The users gender which also can be used to make recommendations based on the partitioning the audience into male and females.</p> <p>fk_city_id : which acts as the foreign key in the table and is linked to the country via state_id to provide regional based recommendations.</p> <p><i>primary_key(user_id) , foreign_key(fk_city_id)</i></p> |
| book_properties | <p>The book_properties consists of columns that describe the books in the book_titles table.</p> <p>Attributes : 1) isbn</p> <p>2)language_id :The id that uniquely identifies the book language and can be further used to make recommendations based on the language of the book.</p> <p>3) is_award_winning : the field is set to zero, by default if the is_award_winning parameter is not 1 and can help to identify the books which have won awards so that we can make a recommendations only on the achievers.</p> <p>4) published_date : The published_date attribute can be used to provide top recommendations on the books</p> |

| | |
|--------------------------|---|
| | <p>based on various eras like the classic old books or top-rated books in the recent years.</p> <p><i>primary_key(isbn) , foreign_key(language_id)</i></p> |
| User_rating | <p>The user_rating table has a entry for each book which the user rates .This ratings form the central parameter toward making the top-rated recommendations .</p> <p>Attributes : 1) isbn : the book for which rating is made.</p> <p>2)fk_user_id: the id of the user who rates the book</p> <p>3) ratings : ratings between the scale of 1-5 , for the user_id and the isbn</p> <p><i>primary_key(isbn, fk_user_id)</i></p> |
| Purchase | <p>The purchase table has the entry for each transaction the user makes . The information about the books purchased per transaction can be found In the 'book_titles_has_purchase' table since a transaction or purchase can have multiple items that are purchased in the same transaction .This table can be used to provide recommendations based on the top popular books In the market.</p> <p>Attributes : 1)purchase_id : A unique id per transaction is generated which is auto_increment.</p> <p>2)fk_user_id : The id of the user who makes the purchase.</p> <p>3) purchase_date : the date on which purchase is made, which can be used to estimate the customer purchase cycles and can in turn be used to send out specific e-mails to the reader to make purchases.</p> <p><i>primary_key(purchase_id), foreign_key(fk_user_id)</i></p> |
| Book_titles_has_purchase | <p>This table is created due to many-to-many relationship that exists between purchase and book_titles. i.e in one</p> |

| | |
|--|---|
| | <p>transaction a user can buy multiple books.</p> <p>Attributes :1) book_titles_isbn – isbn of the purchased book 2) purchase_purchase_id – purchase_id for the transaction.Both the attributes are foreign keys of book_titles and purchase table respectively.</p> <p><i>primary_key</i>(book_titles_isbn, purchase_purchase_id)</p> |
|--|---|

Important factors considered while designing the schema of the database :

- 1) The ‘user-rating’ table as a separate entity rather than clubbing it with the purchase table.

user_rating is one of the driving factors in providing recommendations. The user -rating table is separately made because if the ratings are kept in the purchase table, it would have reduced the querying overhead but at the same time would have increased the space that’s being used because for every purchase irrespective of whether the user rates or doesn’t a space is reserved which is one of the many drawbacks of the relational schema in contrast to non-relational schema which can create columns dynamically. Also a user necessarily need not rate, at the time of purchase without the knowledge of the content and also its often observed that the ratings provided are usually very sparse in contrast to even the purchases being made.

The association between various table in the database is one of the important paradigms that should be kept in mind in designing a well-structured and efficient database model.

3. Database structure: A Normalized View

The database schema consists of tables that act as object or entities of the schema. This data that is made available on the database is organized in a structural pattern of rows and columns. A single row in the column is unique in its entirety while a single column represents the attribute of the entity. For example : In the purchase table each transaction is unique having a purchase_id,

user_id and the date of purchase and comprises of a single row.

The structural integrity of the database is the terminology which aims at validating, and maintaining the correct and accurate data in the database. This structural integrity can be enforced by using different types of integrity constraints.

There are mainly five types of constraints:

1. Domain constraints
2. Unique Key constraints
3. Not Null constraints
4. Referential integrity constraints
5. Semantic integrity constraints

1) Domain constraints:

The Domain constraints defines the range of values and types for the attribute of tuples. In relational database, the domain constraints have to implement as a referential integrity constraint for enumerated values of an attribute because enumerated values can be stored as separate, single-attribute relation (Paul W.P.J. Grefen and Peter M.G. Apers, 1993). For example, Student table has ID as integer and name as a character then the SQL statement as below:

```
Create table user (user_id int (5), name varchar (20))
```

2) Not Null constraints

The Not Null constraints can be observed as a special scenario of domain constraints as they restrict the domain of the attributes. The null vales are not permitted for the given attributes (Paul W.P.J. Grefen and Peter M.G. Apers, 1993). For example restrict the primary key to have null values using the Non-null constraint. The primary key field cannot contain NULL.

```
Create table user(user_id int (5) NOTNULL, name varchar (20))
```

3) Unique Key constraints

It specify the attribute or combination of attributes which should be unique to allow the key to be used to identify the exact tuples. For example user_id must be unique for every user. The non primary key attributes can be specify as unique (Us and Kulkarni, 2007) .

4) Referential integrity constraints

The referential integrity constraints defined as any column of base table declared values from primary key of parent table or null. This key is considered as foreign key of the parent

Table (Work, 1991).

5) Semantic integrity constraints

The semantic integrity constraints provides general restrictions on the data and changes in it. For example, table user_rating has column ratings which should be between 1 to 5.

Categories of Integrity Constraints:

I. ENTITY INTEGRITY

1. PRIMARY KEY

2. UNIQUE KEY

II. REFERENTIAL INTEGRITY

3 FOREIGN KEY

III. DOMAIN INTEGRITY

4. DEFAULT

5. NOT NULL

6. CHECK

IV. USER DEFINED INTEGRITY

7. RULE

Normalization:

Normalization in the database can be defined as the process that (Kumar, 2017) involves decomposition of the database ,into one or many tables to address the problem of data redundancy. The normalization process also involves (Kumar, 2017) in identifying the relationships that exists in the schema.

First Normal Form :

The relational database is assumed to be in the 1nF if there exists no redundancy and also no presence of the multivalued attribute within the database. In our schema the author, themes and qualities can be multiple pertaining to a book that is for a single ISBN, but all the tables are separated out . for example if a a book has multiple authors the, each author is given a rank in correspondence to the authors contribution to the book and stored in a separate table where isbn

and author rank act as a primary key for that table.

Each table possesses a unique primary key which uniquely can identify

The problem of data redundancy is also solved as the user table for example has no attribute of age, because age can be derived from the users date of birth.

Hence we can say that our database is in 1st Normal Form.

Second Normal Form :

For the database to satisfy the 2nd Normal form the database should be in the 1st Normal form and at the same time all the tables in the database there shouldn't exist partial dependencies. i.e. all the attributes in the table are dependent only on the primary key in the table and not on any other column for its existence.

For example : In our database if we consider the themes table the 'theme_name' which is non-key attribute is directly dependent on the (isbn,theme_rank) which together are our primary keys and theme_name column is not at all dependent partially on theme_rank or either isbn.so it satisfies the 2nf form .

Third Normal Form:

For the database to be in the third normal form there should not exist any transitive dependencies in the database and at the same time should satisfy the 2nd Normal Form.

For example : In the users table we have column 'fk_city_id', which depend on the primary_key 'user_id', but we don't have the 'state_id' in the same 'user' table , and have it in the different 'city' table therefore we can say that, it satisfies third normal form. If we had the 'state_id' in the same table, the non-prime column 'city_id' would have determined another non-prime attribute .which is taken care of in our database , hence it is in the third normal form.

Boyce Codd Normal Form (BCNF):

For the database to be in the BCNF form, it should satisfy the 3rd Normal Form, and at the same time for every functional dependency in the form of $X \rightarrow Y$, X is such that it's the super key in the table, we can infer from our ER- diagram that our table satisfies the BCNF form as all the tables consists of a super-key which defines all the attributes in the table.

4. Database Views

The views in the database are nothing but the virtual table created by querying on one or more tables to have access to frequent data. The query to the views is made just like a user queries on the normal relational table. All the modifications on the dataset should be reflected in the view. The view creates a level of abstraction by giving a sense of security, as it doesn't expose the confidential data in the table by providing an interface to query on the created view.

For the scope of 'online Book Recommendation' the following views have been created on top of the existing tables to enable quick processing of the recommendations to be made.

Non-personalized Recommendation based on the identified target audience.

1. A view on the award winning books :

Recommendations are basically provided taking into consideration the top ratings, popularly liked, award winners etc. All the views in the system are created keeping this basic criteria into consideration.

There are certain books which get nominated and amongst which a few are awarded the best winning books at various events. A view I created to provide the award winning books in the database by creating a join on the book_properties and the titles table. So this view will provide a quick recommendations to all the award winning books in the database.

```
CREATE VIEW award_winning_books AS
SELECT DISTINCT
    (t.title)
FROM
    titles t
JOIN
    book_properties bp ON bp.isbn = t.isbn
WHERE
    bp.is_award_winning = 1;
```

Figure 2: view for award winning books

2. A view on classic_old_books and new_books

A view is created on the basis of the top ratedm but old books, this helps in catering to the **target audience** which has a taste for old writings rather than the books written in the modern era. So two views are created taking into consideration the target audience which are driving source for any online system.

The classic old books are recommended by selecting the books which are top-rated, by many users by calculating the average ratings for the books based on the ratings column is the user-rating table and then selecting only the books which have the published date, greater than a specified date.

```
SELECT title from titles where isbn in (SELECT isbn as filtered_isbn
#user_ratings, top_rankers
FROM (SELECT isbn, RANK() over(ORDER BY t1.user_ratings DESC) as top_rankers,user_ratings from
(SELECT
AVG(ur.ratings) AS user_ratings, bp.isbn,bp.published_date
FROM
book_properties bp
JOIN
user_rating ur ON ur.isbn = bp.isbn
GROUP BY ur.isbn having bp.published_date < YEAR(NOW())-9 )t1)t2 where t2.top_rankers between 1 and 5);
```

Figure 3: View on the old- classic books

For creating the above view the Rank() is calculated on the basis of the ratings of the books which is calculated by taking the average of all the user_ratings for that books and later all the books published after a date are recommended for which MySQL Year(Now()) functions are used. A view for top-books in the modern era is created on the similar lines as the old-classic view.

3. Top – recommendations based on the age-group:

There are certain books which are read by children or rather are favorite topics only amongst a particular age group. The recommendation system must take into consideration all this things into consideration while making recommendations .As target audience when provided with proper recommendations more is the probability of increasing the sales.

The following view is created based on selecting the top rated books but only the ones which are popular amongst the children or adults .

The recommendations based on the age- group are provided keeping in mind a certain assumed threshold date (date of birth) which distinguishes the adult from the children. The age column is purposefully **not created** in the user table because, it can **always be derived** from the date of birth and also prevents **indirect duplication of the data**.

| | title | user_ratings | top_rankers |
|---|---------------------------------------|--------------|-------------|
| ► | Emma | 5.0000 | 1 |
| | Room | 4.0000 | 2 |
| | The Three Musketeers | 3.0000 | 3 |
| | Midnight's Children | 3.0000 | 3 |
| | Percy Jackson and the Lightning Thief | 2.5000 | 5 |

Figure 4: Output for query on top rated books for children

4. **Top – recommendations based on gender :**

A view is also created based on the top male and female recommendations on the top rated books in the respective gender.

| | title | user_ratings | top_rankers |
|---|---|--------------|-------------|
| ► | Room | 4.5000 | 1 |
| | A Game Of Thrones | 3.5000 | 2 |
| | Concepts in Bioinformatics and Genomics | 3.0000 | 3 |
| | Percy Jackson and the Lightning Thief | 3.0000 | 3 |
| | The Adventures of Sherlock Holmes | 2.7273 | 5 |

Figure 5: Output for query on top rated books for Males

5. **Views for popular books :**

A view for popular books is created based on the number of copies of the books that are sold . This view is created by querying on the intermediary table that is 'book_titles_has_purchases' that contains the isbnns of all the books and the 'purchase id'.


```

create view popular_sellers as SELECT title from titles where isbn IN (SELECT book_titles_isbn
FROM (SELECT *,
      Rank ()
      OVER (
        ORDER BY t1.copies_sold DESC) AS popular_isbns_rank
FROM (SELECT book_titles_isbn,
      Count(book_titles_isbn) AS copies_sold
FROM book_titles_has_purchase |
GROUP BY book_titles_isbn)t1) t2 where t2.popular_isbns_rank between 1 and 5);

```

Figure 6: Query to create a view for Popular sellers

6. Views for Top-rated books :

A view for the top-rated books is created based on the the 'reader ratings' of the books, the average rating of the books is considered for this. It calculated by joining the titles and the user rating table.

The output of the 'top-rated books' view.

| | title | user_ratings | top_rank |
|---|---|--------------|----------|
| ▶ | A Game Of Thrones | 3.8889 | 1 |
| | Concepts in Bioinformatics and Genomics | 3.6667 | 2 |
| | Room | 3.6667 | 2 |
| | Emma | 3.3636 | 4 |
| | The Three Musketeers | 3.2222 | 5 |

Figure 7: output for top rated books

Both the views created in pointers 5 and 6 are later used in unison to provide the recommendations, which is explained in the 'Databases in Action' part of the report.

5. Procedural Elements

Procedural elements can be thought of routines which contains a sequence of steps that are to be executed in order to get the desired tasks done that can be called from other procedures. The stored procedure helps to improve the modularity of the code and hence can be reused again . The stored procedures also helps to maintain data integrity by providing a layer of abstraction between the end user and the database.

The book recommender system database makes use of all the three types of procedural elements : 1) Triggers 2) Stored procedures and 3) Functions

1) Triggers :

The triggers have been implemented at two at two stages for the design in the database schema.

1) 'Validate_gender' trigger

A trigger on insert for the gender attribute , in the user table to store the standard character for the genders M for males and F for Females in the database.

2) 'Validate_language' trigger

A trigger on insert for the language_name attribute, in the bok_language table to store the standard acronyms for the specific language.

```
delimiter //
create trigger gender_validation BEFORE insert on `user`
FOR EACH ROW
BEGIN
IF NEW.gender = "male" or NEW.gender = "MALE" THEN
    SET New.gender="M";
ELSEIF NEW.gender = "FEMALE" or NEW.gender = "female" THEN
    SET New.gender="F";
END IF;
END; //
```

Figure 8: Gender Trigger

2) Functions :

MySQL allows the creation of user defined functions, that accept n-number of parameters as an input but have a single-valued output (Source:<https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html>). Function thus can be used to reduce the boiler-plate code and at the same time, they also can be used in the SQL statements and the stored procedure (Source:<https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html>), which also helps in efficient code readability. Function have been implemented

to get the value of 'country_id' from the 'country_name' attribute which is used multiple times. The function also helps in the dynamic input(user-defined) of values in contrast to changing the query for each new-value that arrives.

```
CREATE DEFINER='root'@'localhost' FUNCTION `get_country_id`(countryName varchar(255)) RETURNS int(11)
    DETERMINISTIC
BEGIN
    DECLARE cont_id int(11);
    SELECT country_id INTO cont_id FROM country WHERE country_name = countryName;
    RETURN cont_id;
END
```

Figure 9: get_country_id function

The above function takes parameter countryname as the input and returns country_id which is an int datatype, all this is specified in the method declaration followed by the type of routine it is. A routine can be 'Deterministic' OR 'NOT DETERMINISTIC'. The 'Deterministic' states that the function will return the same output for every call for the same set of inputs while 'Not Deterministic' doesn't guarantee the same output even for the same set of inputs.

A call to the function is made using the following statement :

```
select books.get_country_id('USA');
```

3) Stored procedures :

A stored procedure is similar to that of a function but doesn't have a return statement. The procedures also accept n-number of parameters as an input. The important point to note about the stored procedure is that it cannot be used in the SQL statements unlike the functions.

In the Book Recommender system the procedures have been implemented to recommend books based on the language, region of the user also based on user-preferences for a personalized recommendation .

The following three stored procedures are implemented :

- 1) The store proc that gives top recommendation of the books based on the particular language by taking in the language name as the input parameter
- 2) The stored procedure that gives top recommendation of the books based on the particular region by taking region name as the input parameter.
- 3) The stored procedure that gives a list of user-based recommendation of the books by recommending top books that the user hasn't purchased.

All the above stored procedure are a necessity and need to be implemented for the book recommendation system as all deal with a complex set of steps and more importantly they consist of the parameter that can be dynamically varied as per the reader requirement and provide prompt recommendations.

The below stored procedure, the countryName is taken as input and further computations are made to calculate the top rated books in that particular region. The language based recommendation stored procedure is defined in the similar manner as the region_based_recommendation.

```
CREATE DEFINER='root'@'localhost' PROCEDURE `region_based_recommendations`(IN countryName varchar(255))
BEGIN
SELECT title,
       user_ratings, top_rankers
FROM (SELECT title, RANK() over(ORDER BY t1.user_ratings DESC) as top_rankers,user_ratings from
(SELECT
  AVG(ur.ratings) AS user_ratings, t.title
FROM
  user u
  INNER JOIN user_rating ur on ur.fk_user_id = u.id
  Inner join titles t on t.isbn = ur.isbn
  where u.fk_city_id in (SELECT
  cit.city_id
FROM
  country cont
  INNER JOIN
  state st ON cont.country_id = st.fk_country_id
  INNER JOIN
  city cit ON st.state_id = cit.fk_state_id
WHERE
  cont.country_id IN (select books.get_country_id(countryName)))
GROUP BY ur.isbn )t1)t2 where t2.top_rankers between 1 and 5;
END
```




Figure 10: Stored procedure for recommendation based on region

(Function call from the stored procedure)

The following is the output of the stored-procedure for country based recommendations.

| | title | user_ratings | top_rankers |
|---|---|--------------|-------------|
| ► | The Three Musketeers | 3.5714 | 1 |
| | Emma | 3.5000 | 2 |
| | Concepts in Bioinformatics and Genomics | 3.5000 | 2 |
| | Room | 3.0000 | 4 |
| | Percy Jackson and the Lightning Thief | 2.5714 | 5 |

Figure 11: Stored procedure output for recommendation based on region

2.stored procedure for personalized reader recommendations.

The other important procedure is to calculate the user recommendation by recommending only the top rated books which are not purchased by the user thereby making the personalized recommendation for the user. In order to calculate this the ratings of all the books are calculated based on the user ratings for the movies (average of all the ratings for that particular movie) , after which the only books which are recommended are the ones which are not purchased by the user .

```

CREATE DEFINER=`root`@`localhost` PROCEDURE `top_recommendations_for_user`(IN userId INT)
BEGIN
SELECT
    title
FROM
    titles
WHERE
    isbn IN (SELECT
        t1.filtered_books AS isbn FROM
        (SELECT DISTINCT
            (isbn) AS filtered_books, AVG(ratings) AS top_ratings FROM
            user_rating GROUP BY isbn
            HAVING top_ratings > 3) t1
        WHERE
            t1.filtered_books NOT IN (SELECT DISTINCT
                (bp.book_titles_isbn)
            FROM
                purchase p
                JOIN
                book_titles_has_purchase bp ON bp.purchase_purchase_id = p.purchase_id
            WHERE
                p.fk_user_id = userId));
END

```

Figure 12: Stored procedure for personalized reader recommendations

The call to the above stored procedure is made based on the following statement
call books.top_recommendations_for_user(2);

The output of the above recommendation is in the following form:

| | title |
|---|---|
| ▶ | The Three Musketeers |
| | Room |
| | Concepts in Bioinformatics and Genomics |
| | Emma |

Figure 13: Personalized user recommendations

6.Example Queries : Your database in action.

1.Top – recommendations based on ratings ‘top_rated_books’ and the ‘popularly selling books’:

Recommendations cannot only be made based on the top- rated books they can also be made on the top-sellers that for example: consider a book ‘Data science in Python’ it may be top-rated but it cannot be

recommended only because its top-rated to any use who isn't interested in the scientific themes and in the similar manner 'Game of thrones' just because it is popular and sold more cannot be recommended to a user for its popularity. The recommendations in all should consider diverse, popular and top-rated recommendations . so **two** views are created as shown in the section-4 ,based on the books which have been sold more – 'popular sellers' and 'top_rated books' based on the user_ratings. Now we can create a union of the above views to give the top recommendations to the user. The following will be the output for recommended books(Non-personalized)

| top_recommended_books |
|---|
| The Adventures of Sherlock Holmes |
| A Game Of Thrones |
| The Three Musketeers |
| Women Fire and Dangerous Things |
| Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies |

Figure 14: Non-personalized : Top Recommended Books

2. Top – recommended books based on authors, themes and qualities.

Two view for each top authors, themes and qualities is created based on the popularity and ratings for the books. These are later utilized to recommend books on the basis of authors, themes and qualities respectively.

```
SELECT title from titles where isbn in (
    select isbn from themes where theme_name in (
        select theme_name from top_rated_themes where top_rankers < 3
        union
        select theme_name from popular_themes where popular_theme_rank_in_trending_books < 3))
limit 5 offset 0;
```

Figure 15: Query for top recommender's author

The above query select the books for recommendations on the top themes which are selected as unison of both popularity and ratings. We can see

that union operator is used to select the top-themes from two views . The recommendations are limited for 5 and start from 0- offset as the user would like to have more recommendations the offset can be increased in the multiples of 5, to give the user more recommendations. A similar set of query can be executed on the views for authors and qualities to provide such recommendations.

| | Top_rated_books_based_on_Themes |
|---|---------------------------------------|
| ► | Where Eagles Dare |
| | The Lord Of The Rings: The Two Towers |
| | A Game Of Thrones |
| | And Then There Were None |
| | The Murder of Roger Ackroyd |

Figure 16: Books based on top themes

3. Top – recommended books for a user based on the type of books he has read .

A personalized recommendation is provided on the basis of books he has read.

- a. All the books purchased by user are not recommended by using the not in query.
- b. The top-rated themes for a user are selected, by selecting the theme based on the theme_rank i.e. if the theme_rank is less than 2 than only that theme_name is selected.
- c. The books are now recommended based on the themes i.e. books which contain the top-themes for that user and also the books which the user has purchased are not recommended.


```

select title from titles where isbn in (select t2.theme_isbns as theme_isbns from (SELECT isbn as theme_isbns FROM themes
WHERE
    theme_name IN (SELECT t1.theme_name
        FROM (SELECT DISTINCT
            (theme_name), theme_rank FROM themes
        WHERE isbn IN (SELECT isbn FROM user_rating
            WHERE ratings >= 3 AND fk_user_id = 14)) t1
WHERE
    t1.theme_rank < 2))t2 where theme_isbns not in (SELECT DISTINCT
    (bp.book_titles_isbn)
FROM
    purchase p
    JOIN
    book_titles_has_purchase bp ON bp.purchase_purchase_id = p.purchase_id
WHERE
    p.fk_user_id = 14))

```

Figure 17: Books based on top themes for that user(personalized)

| title |
|---|
| Concepts in Bioinformatics and Genomics |
| Introduction to Bioinformatics |
| Metaphors We Live By |
| Women Fire and Dangerous Things |
| Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies |
| Applied Bioinformatics: An Introduction |

Figure 18: output for -Books based on top themes for that user(personalized)

7.Conclusion

The Recommender system help in providing relevant information in a timely manner to the end user , from all the extensive digital information present. The project on the 'online Book Recommendation' system at large is implemented to provide 'non-personalised' and 'personalised' recommendation to the user, which are two important paradigms of any recommendation system. The collaborative filtering which is a part of personalized recommendations and content-based filtering which depend on the comparison of user and item profiles (Souce:<http://recommender-systems.org/content-based-filtering/>) is implemented together, hence the implemented system, can be termed as the hybrid recommender system.

The project can be further extended by providing a user- search history table

that can be crawled to provide more refined and updated recommendations to the users based on the search history of the user, and other users can also be benefited as they would be recommended the books based on the trending topic amongst all the users in the system. The another advancement to the project can be collection of more user-relevant information from the various API's and social media platforms like Facebook, user tweets and storing them in the tables to collect the topics user is interested in and then make recommendations based on it.

The project thus formulated important concepts from designing a normalized database, advanced querying, making use of PLSQL and also important concepts related to the topic of recommendations system which are a necessity in the digital world.

8. Acknowledgements

I would like to express my sincere appreciation to my prof. Tony Veale for his academic support in Relational Databases, throughout the semester . I would also like to thank all the Teaching Assistants for their continual help and support during the entire coursework and clearing my doubts. Also, I would like to thank University College Dublin for providing me with an innovative platform for showcasing my ideas and abilities.

References

Content Based Filtering, <http://recommender-systems.org/content-based-filtering> [Date accessed:26/05/2019]

Dr. Michael O'Mahony (no date) 'Recommender Systems & Collective Intelligence Introduction & Module Overview'.

ER Diagram, <https://searchdatamanagement.techtarget.com/definition/entity-relationship-diagram-ERD> [Date accessed:02/05/2019]

Isinkaye, F. O. (2015) 'Recommendation systems : Principles , methods and evaluation'. Ministry of Higher Education and Scientific Research, pp. 261–273.
doi: 10.1016/j.eij.2015.06.005.

Kumar, K. (2017) 'Relational Database Normalization under Tabular Approach : A Design Methodology', 8(5), pp. 2160–2165.

Paul W.P.J. Grefen and Peter M.G. Apers (1993) 'Integrity control in relational database systems- An overview', 10, pp. 187–223.

Procedure vs Functions in mysql,
<https://dev.mysql.com/doc/refman/8.0/en/create-procedure.html> [Date accessed: 01/05/2019]

Us, C. A. and Kulkarni, S. D. (2007) '(12) United States Patent', 2(12).

Work, R. (1991) 'Lawrence Berkeley National Laboratory'.