USE CASE STUDY REPORT

Group no: 04

Student Names: Arvind Garg and Malavika Krishnan

Executive Summary:

The objective of this study was to design and implement a production ready relational database that can be implemented in a real world Appstore environment. With the exploding number of mobile users, fault tolerant and reliable application distribution architectures become essential. The database used in this study was modelled by analyzing existing Appstore architecture like Apple's Appstore and Google's Playstore. All the requirements were carefully collected to develop the ER and UML diagrams. These conceptual models successfully modeled the use case requirements. They were then mapped to a relational model where the requirements and constraints were satisfied by using combinations of primary keys, foreign keys and relations. Moreover, the relational model was normalized to reduce redundancy and increase reliability. This database was then fully implemented in MySQL and NoSQL.
The use of incorporating database connections with R studio helped to get quality insights from the dataset overall. In the future, the functionality of this DBMS can be enhanced by including more querying options and incorporating complex relationships and entities.

I. Introduction

Apple is in need of a new database system for its Appstore for the new version of iOS. In the Appstore, developers publish their pre-approved apps. Alongside the existing purchase model, the new Appstore should also support the new subscription-based model introduced by Apple where users can subscribe for applications. An application can be of three types – a free app, a paid app or an app with a subscription model. The Appstore should allow the user to download any of these three types of applications. The new Appstore should also limit users to be able to download apps available in their country. Apple earns a 30% cut off the app price and the rest goes to the developer. Users should be able to review an application by giving it a rating between 1 to 5 stars and by making a comment.

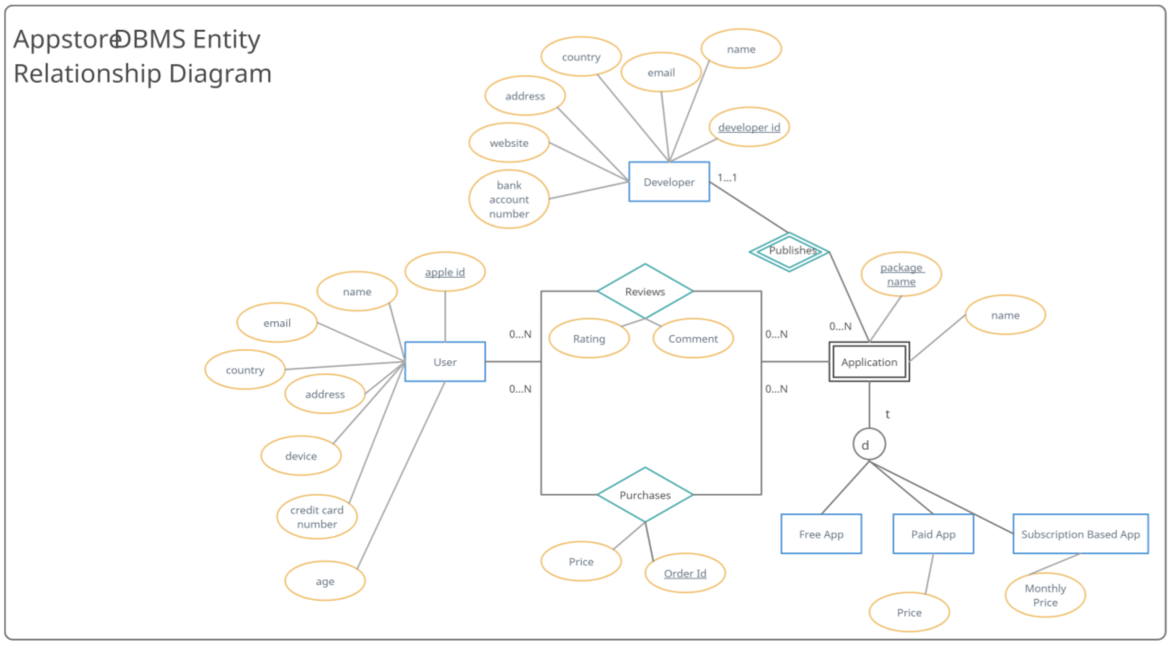
The tables included are:
User_data, Developer, Application, Purchases, and Reviews. These tables are to be created in order to successfully meet the requirements of the problem statement.
The database requirements are:

- A developer can publish multiple apps. Each developer has a unique developer id.
- Each application has its unique package name which is used to identify the application.
- Multiple applications can have the same display name, but the package name is always unique.

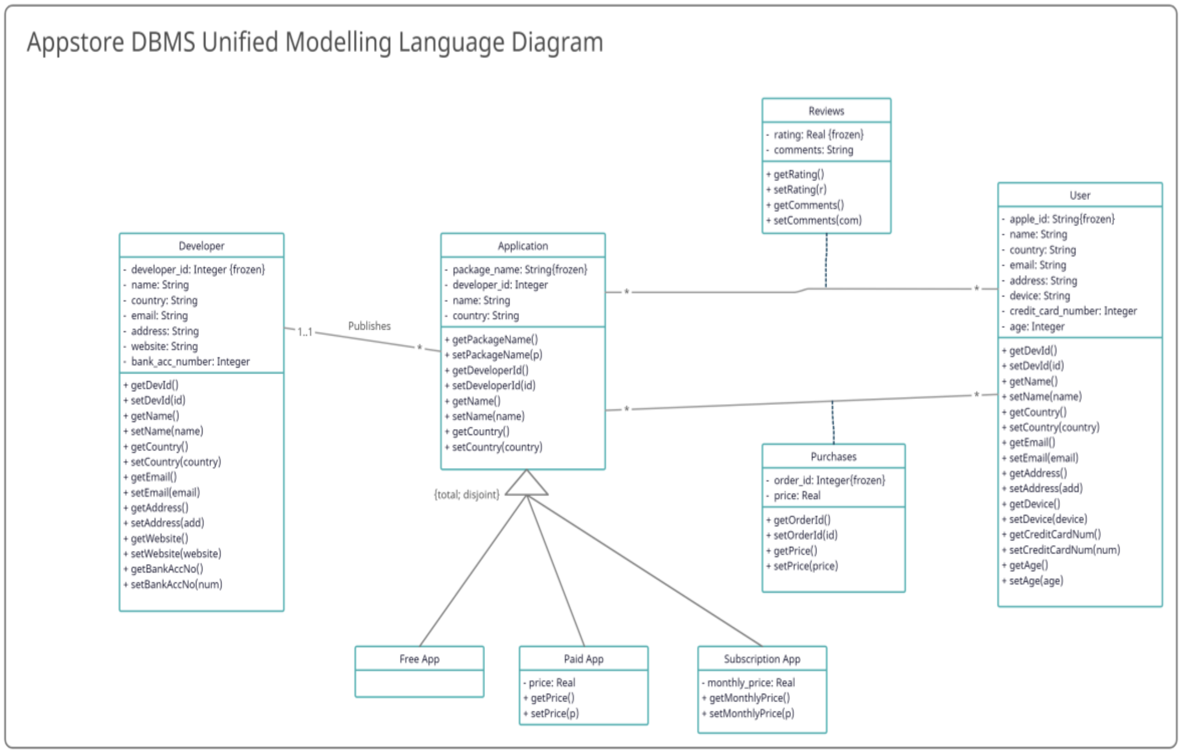
- Users are identified by their unique apple id.
- A user can rate an app between 0 to 5 stars. Users can also provide comments with their rating.
- A user pays with their credit/debit card linked with their apple id.

II. Conceptual Data Modelling

- ### Enhanced Entity Relationship (EER) Diagram



- ### Unified Modelling Language (UML) Diagram



The UML diagram is depicted below in order to demonstrate various versions of the relation captured.

III. Mapping Conceptual Model to Relational Model

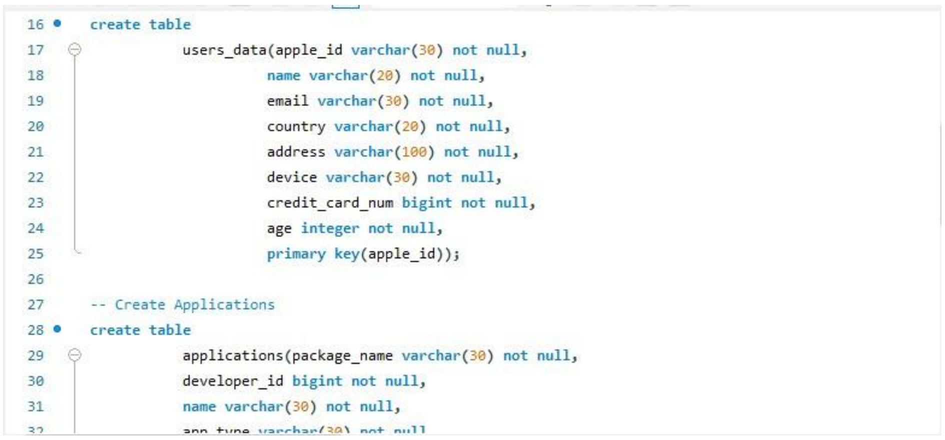
(Primary keys are underlined; foreign key is mentioned in italics; not null values are specified)

- **Developer** (developer_id, name, email, country, website, address, bank_acc_number)
- **User** (apple_id, name, email, country, address, device, credit_card_num, age)
- **Application** (package_name, developer_id, name, app_type, price) app_type constraint will be enforced in the programming language
- **Purchases** (order_id, apple_id, *app*, price, purchase_date)
  Foreign Key app refers to package_name in Application; NOT NULL
  Foreign Key apple_id refers to apple_id in User; NOT NULL
- **Reviews**(user, *app*, rating, comments)
- Foreign Key app refers to package_name in Application; NOT NULL Foreign Key user refers to apple_id in User; NOT NULL

User and app together make up the foreign key of this relation.

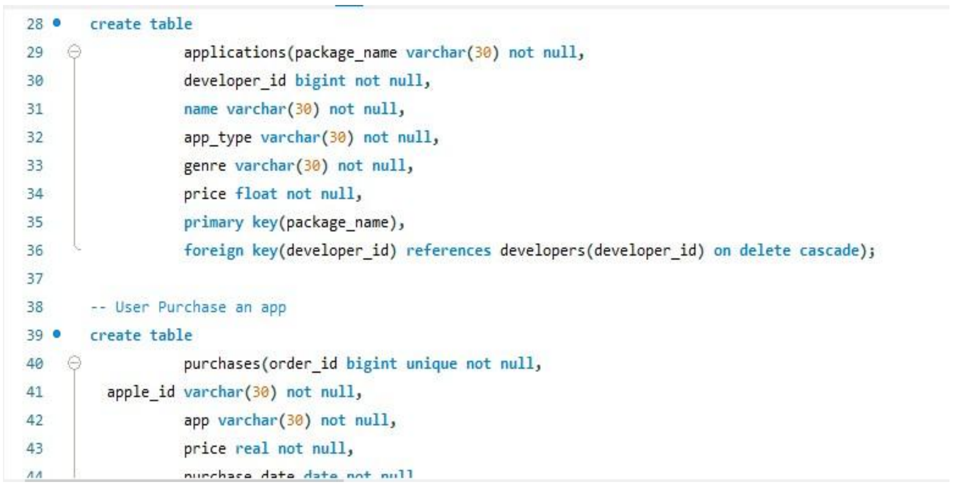
IV. Implementation of Relational Model via MySQL and NoSQL

<u>Relational Model via MySQL:</u>
<u>Table creation:</u>



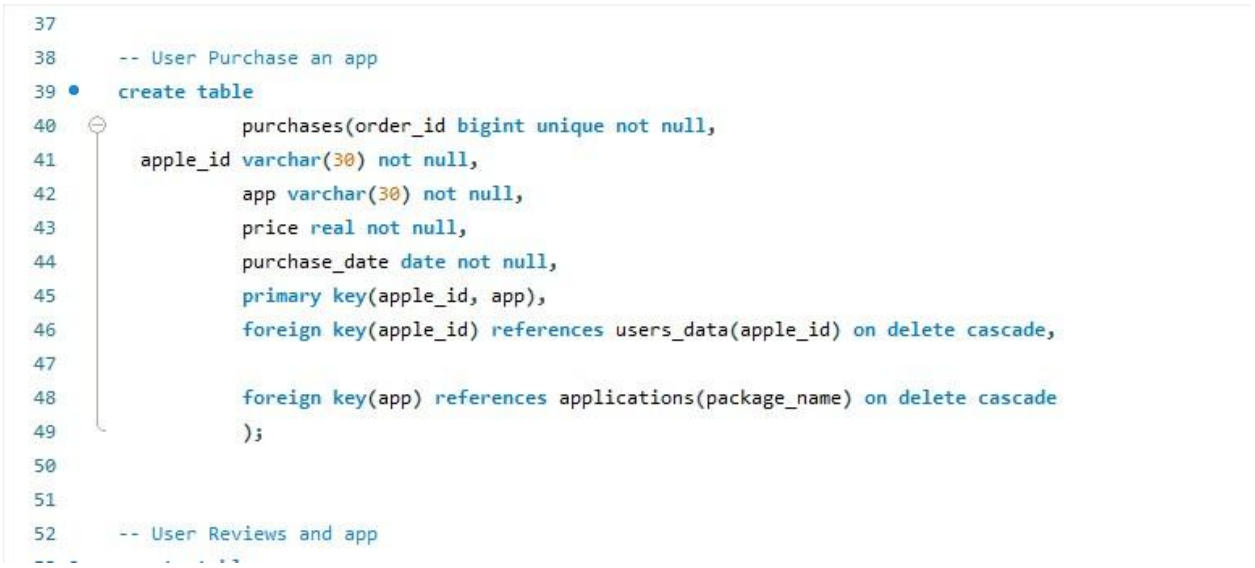
```
create table
        users_data(apple_id varchar(30) not null,
               name varchar(20) not null,
               email varchar(30) not null,
               country varchar(20) not null,
               address varchar(100) not null,
               device varchar(30) not null,
               credit_card_num bigint not null,
               age integer not null,
               primary key(apple_id));

-- Create Applications
create table
        applications(package_name varchar(30) not null,
        developer_id bigint not null,
        name varchar(30) not null,
        app_type varchar(30) not null
```



```
create table
        applications(package_name varchar(30) not null,
        developer_id bigint not null,
        name varchar(30) not null,
        app_type varchar(30) not null,
        genre varchar(30) not null,
        price float not null,
        primary key(package_name),
        foreign key(developer_id) references developers(developer_id) on delete cascade);

-- User Purchase an app
create table
        purchases(order_id bigint unique not null,
  apple_id varchar(30) not null,
        app varchar(30) not null,
        price real not null,
        purchase_date date not null
```

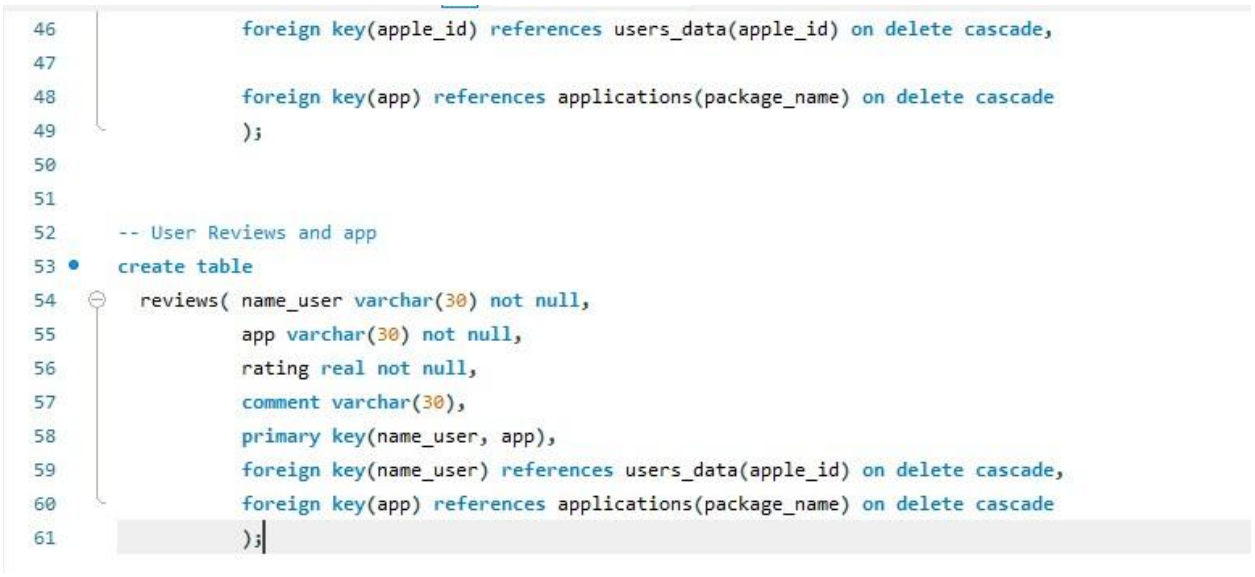


```sql

-- User Purchase an app
create table
          purchases(order_id bigint unique not null,
  apple_id varchar(30) not null,
          app varchar(30) not null,
          price real not null,
          purchase_date date not null,
          primary key(apple_id, app),
          foreign key(apple_id) references users_data(apple_id) on delete cascade,

          foreign key(app) references applications(package_name) on delete cascade
          );


-- User Reviews and app
```



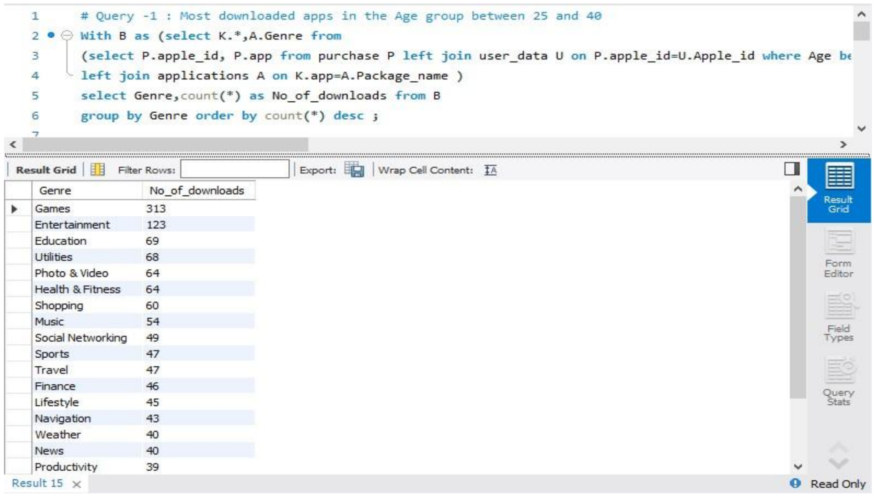
```sql
          foreign key(apple_id) references users_data(apple_id) on delete cascade,

          foreign key(app) references applications(package_name) on delete cascade
          );


-- User Reviews and app
```

```sql
create table
  reviews( name_user varchar(30) not null,
          app varchar(30) not null,
          rating real not null,
          comment varchar(30),
          primary key(name_user, app),
          foreign key(name_user) references users_data(apple_id) on delete cascade,
          foreign key(app) references applications(package_name) on delete cascade
          );
```

Query 1:

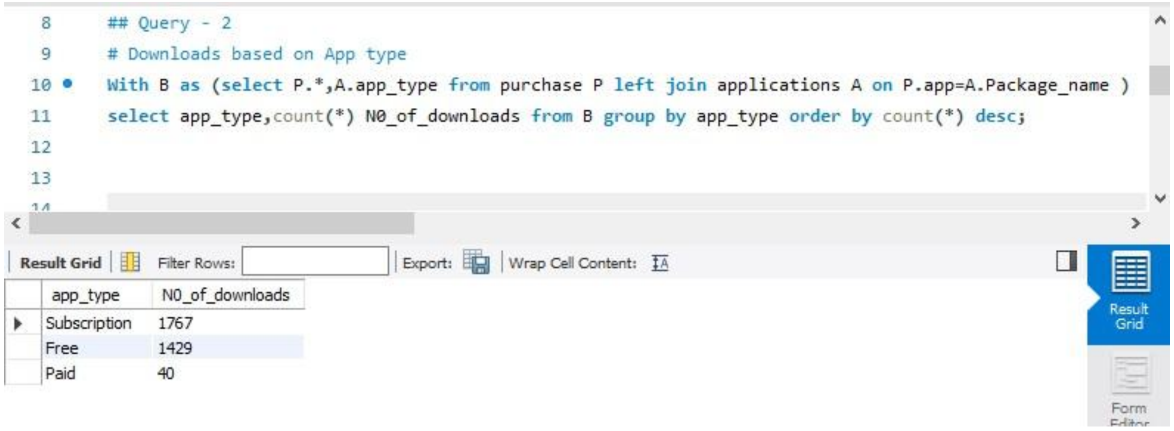


```sql
# Query -1 : Most downloaded apps in the Age group between 25 and 40
With B as (select K.*,A.Genre from
(select P.apple_id, P.app from purchase P left join user_data U on P.apple_id=U.Apple_id where Age be
left join applications A on K.app=A.Package_name )
select Genre,count(*) as No_of_downloads from B
group by Genre order by count(*) desc ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Genre	No_of_downloads
Games	313
Entertainment	123
Education	69
Utilities	68
Photo & Video	64
Health & Fitness	64
Shopping	60
Music	54
Social Networking	49
Sports	47
Travel	47
Finance	46
Lifestyle	45
Navigation	43
Weather	40
News	40
Productivity	39

Query 2:

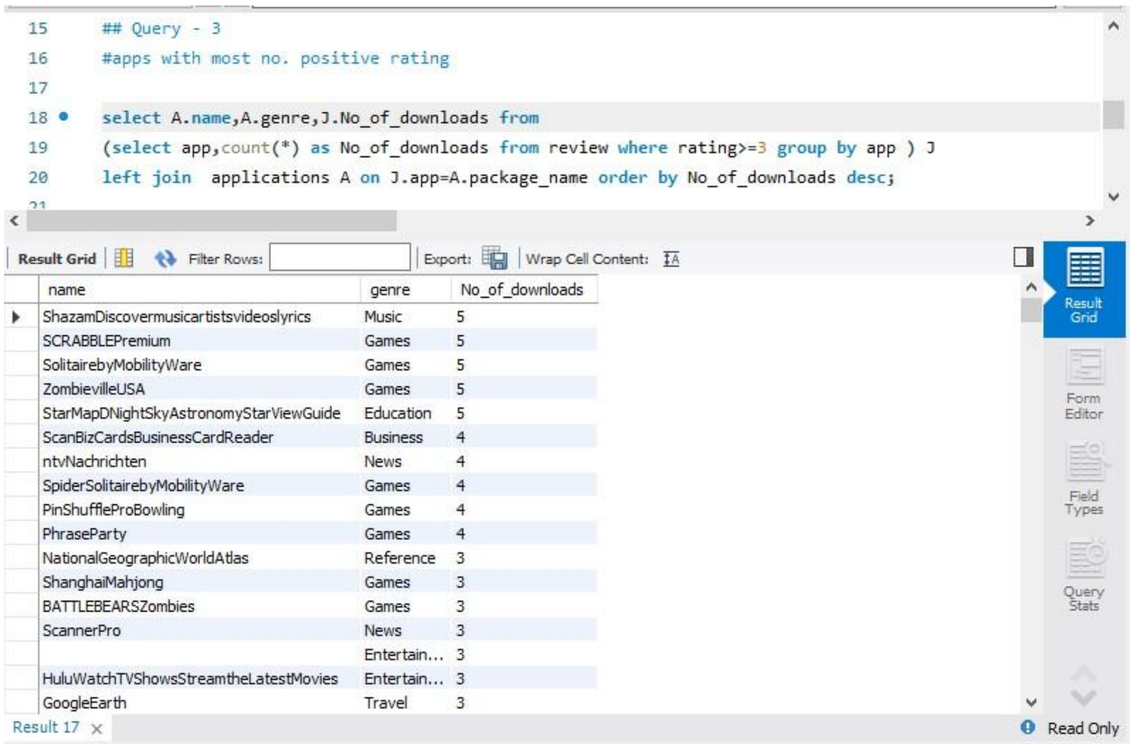


```sql
## Query - 2
# Downloads based on App type
With B as (select P.*,A.app_type from purchase P left join applications A on P.app=A.Package_name )
select app_type,count(*) N0_of_downloads from B group by app_type order by count(*) desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

app_type	N0_of_downloads
Subscription	1767
Free	1429
Paid	40

Query 3 -

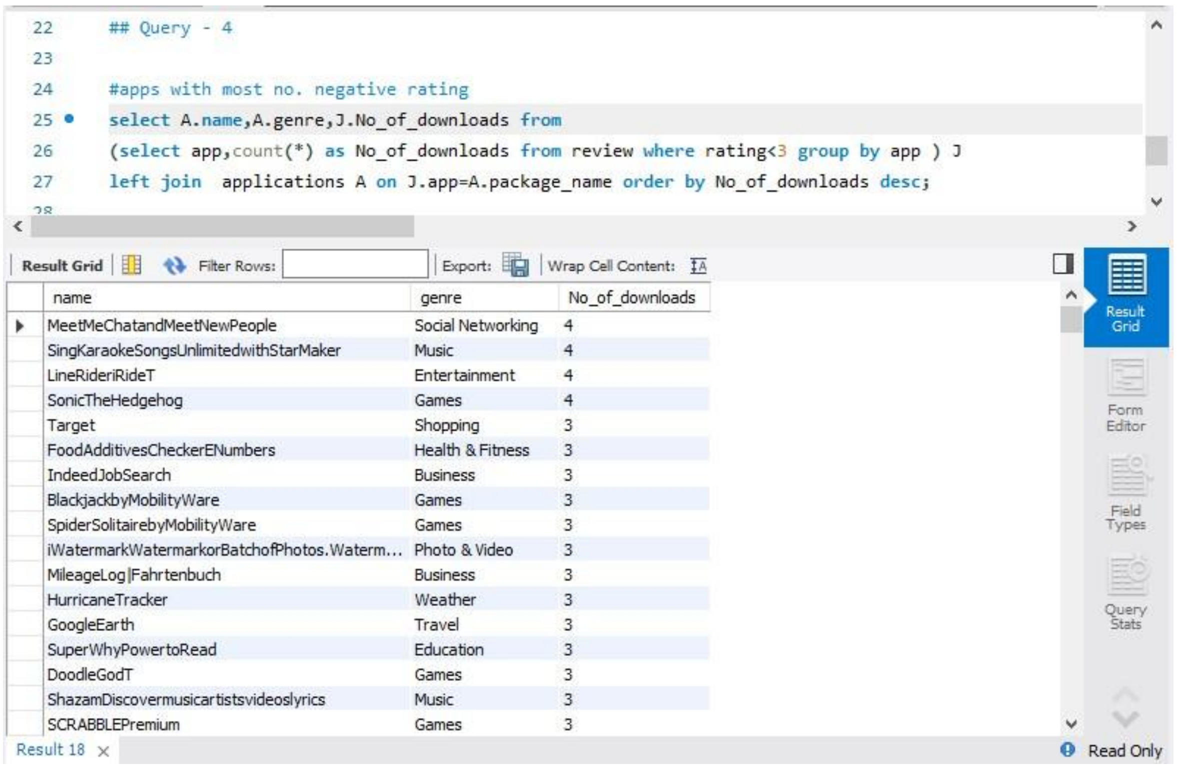


```
## Query - 3
#apps with most no. positive rating

select A.name,A.genre,J.No_of_downloads from
(select app,count(*) as No_of_downloads from review where rating>=3 group by app ) J
left join  applications A on J.app=A.package_name order by No_of_downloads desc;
```

name	genre	No_of_downloads
ShazamDiscovermusicartistsvideoslyrics	Music	5
SCRABBLEPremium	Games	5
SolitairebyMobilityWare	Games	5
ZombievilleUSA	Games	5
StarMapDNightSkyAstronomyStarViewGuide	Education	5
ScanBizCardsBusinessCardReader	Business	4
ntvNachrichten	News	4
SpiderSolitairebyMobilityWare	Games	4
PinShuffleProBowling	Games	4
PhraseParty	Games	4
NationalGeographicWorldAtlas	Reference	3
ShanghaiMahjong	Games	3
BATTLEBEARSZombies	Games	3
ScannerPro	News	3
	Entertain...	3
HuluWatchTVShowsStreamtheLatestMovies	Entertain...	3
GoogleEarth	Travel	3

Query 4 -

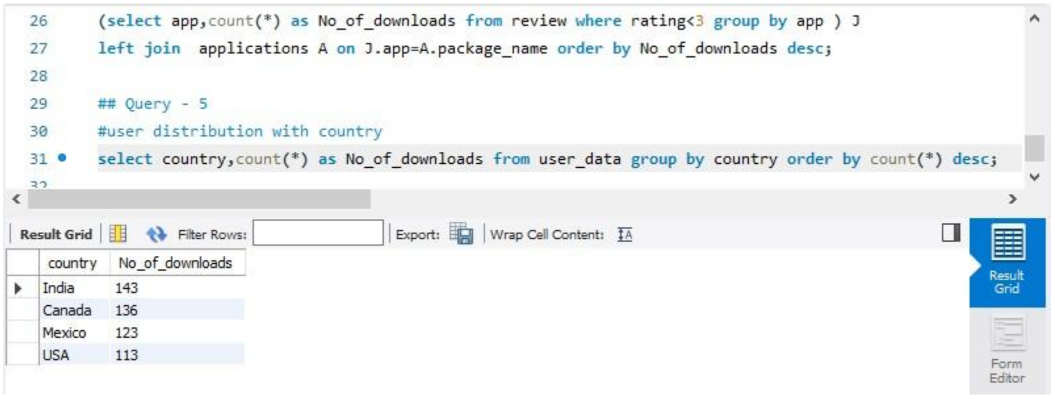


```
## Query - 4

#apps with most no. negative rating
select A.name,A.genre,J.No_of_downloads from
(select app,count(*) as No_of_downloads from review where rating<3 group by app ) J
left join  applications A on J.app=A.package_name order by No_of_downloads desc;
```

name	genre	No_of_downloads
MeetMeChatandMeetNewPeople	Social Networking	4
SingKaraokeSongsUnlimitedwithStarMaker	Music	4
LineRideriRideT	Entertainment	4
SonicTheHedgehog	Games	4
Target	Shopping	3
FoodAdditivesCheckerENumbers	Health & Fitness	3
IndeedJobSearch	Business	3
BlackjackbyMobilityWare	Games	3
SpiderSolitairebyMobilityWare	Games	3
iWatermarkWatermarkorBatchofPhotos.Waterm...	Photo & Video	3
MileageLog|Fahrtenbuch	Business	3
HurricaneTracker	Weather	3
GoogleEarth	Travel	3
SuperWhyPowertoRead	Education	3
DoodleGodT	Games	3
ShazamDiscovermusicartistsvideoslyrics	Music	3
SCRABBLEPremium	Games	3

Query 5 -



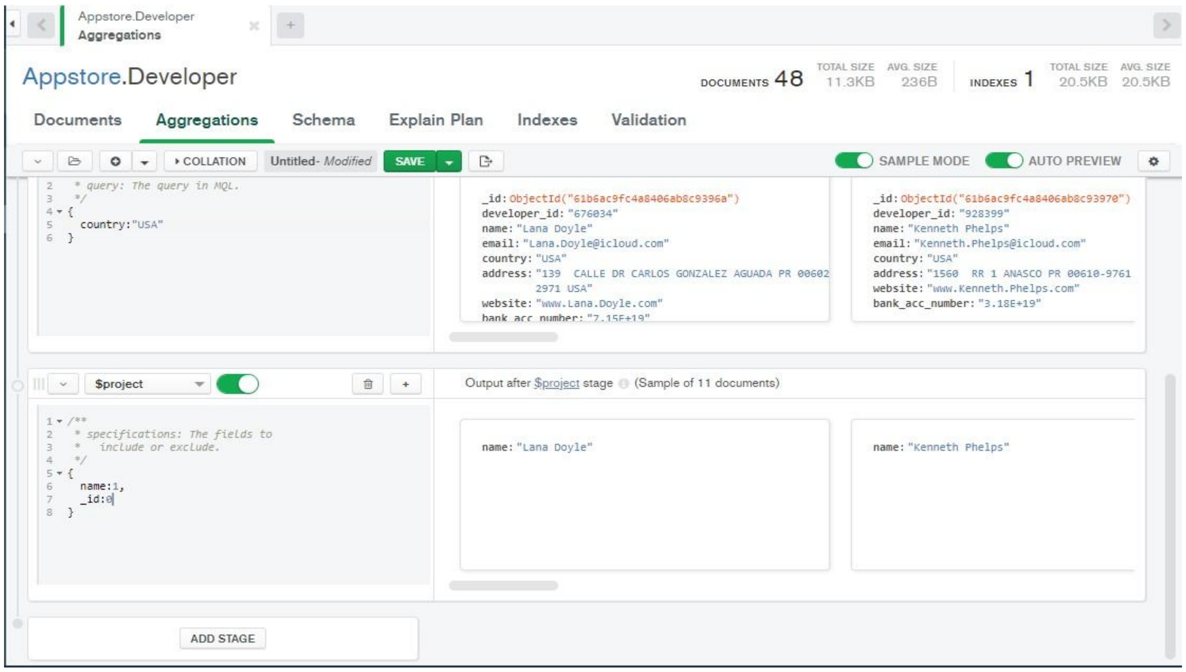
```
(select app,count(*) as No_of_downloads from review where rating<3 group by app ) J
left join  applications A on J.app=A.package_name order by No_of_downloads desc;

## Query - 5
#user distribution with country
select country,count(*) as No_of_downloads from user_data group by country order by count(*) desc;
```

country	No_of_downloads
India	143
Canada	136
Mexico	123
USA	113

<u>Relational Model via NoSQL:</u>

Query 1 - From the developers table we are extracting the name of the developers who reside in USA



Appstore.Developer
Aggregations

Appstore.Developer — DOCUMENTS 48 TOTAL SIZE 11.3KB AVG. SIZE 236B — INDEXES 1 TOTAL SIZE 20.5KB AVG. SIZE 20.5KB

Documents | Aggregations | Schema | Explain Plan | Indexes | Validation

COLLATION | Untitled- *Modified* | SAVE | SAMPLE MODE | AUTO PREVIEW

```
 * query: The query in MQL.
 */
{
  country:"USA"
}
```

```
_id: ObjectId("61b6ac9fc4a8406ab8c9396a")
developer_id: "676034"
name: "Lana Doyle"
email: "Lana.Doyle@icloud.com"
country: "USA"
address: "139  CALLE DR CARLOS GONZALEZ AGUADA PR 00602
        2971 USA"
website: "www.Lana.Doyle.com"
bank_acc_number: "7.15E+19"
```

```
_id: ObjectId("61b6ac9fc4a8406ab8c93970")
developer_id: "928399"
name: "Kenneth Phelps"
email: "Kenneth.Phelps@icloud.com"
country: "USA"
address: "1560  RR 1 ANASCO PR 00610-9761
website: "www.Kenneth.Phelps.com"
bank_acc_number: "3.18E+19"
```

$project — Output after $project stage (Sample of 11 documents)

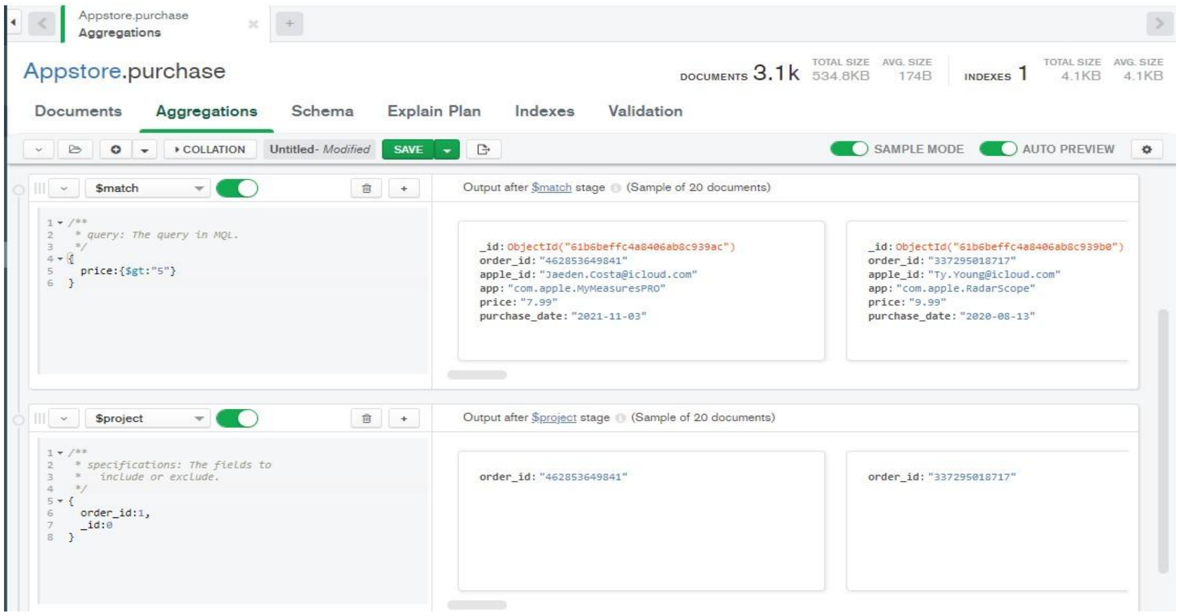
```
/**
 * specifications: The fields to
 *   include or exclude.
 */
{
  name:1,
  _id:0
}
```

```
name: "Lana Doyle"
```

```
name: "Kenneth Phelps"
```

ADD STAGE

Query 2 - From the purchase table we are extracting the order_id with price more than 5 USD



Appstore.purchase
Aggregations

Appstore.purchase — DOCUMENTS 3.1k TOTAL SIZE 534.8KB AVG. SIZE 174B — INDEXES 1 TOTAL SIZE 4.1KB AVG. SIZE 4.1KB

Documents | Aggregations | Schema | Explain Plan | Indexes | Validation

COLLATION | Untitled- *Modified* | SAVE | SAMPLE MODE | AUTO PREVIEW

$match — Output after $match stage (Sample of 20 documents)

```
/**
 * query: The query in MQL.
 */
{
  price:{$gt:"5"}
}
```

```
_id: ObjectId("61b6beffc4a8406ab8c939ac")
order_id: "462853649841"
apple_id: "Jaeden.Costa@icloud.com"
app: "com.apple.MyMeasuresPRO"
price: "7.99"
purchase_date: "2021-11-03"
```

```
_id: ObjectId("61b6beffc4a8406ab8c939b0")
order_id: "337295018717"
apple_id: "Ty.Young@icloud.com"
app: "com.apple.RadarScope"
price: "9.99"
purchase_date: "2020-08-13"
```

$project — Output after $project stage (Sample of 20 documents)

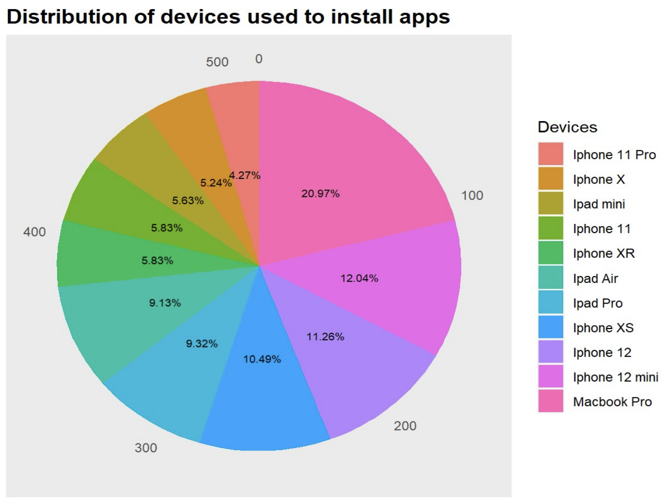
```
/**
 * specifications: The fields to
 *   include or exclude.
 */
{
  order_id:1,
  _id:0
}
```

```
order_id: "462853649841"
```

```
order_id: "337295018717"
```

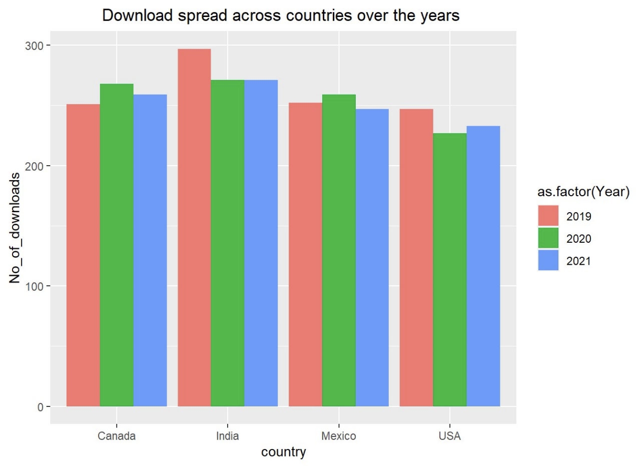
V. Database Access via R

Plot-1:



- This is a pie-chart that represents the distribution of various devices used to install the applications.
- It can be observed that Macbook Pro was the maximum used device to download the apps followed by Iphone 12 mini.

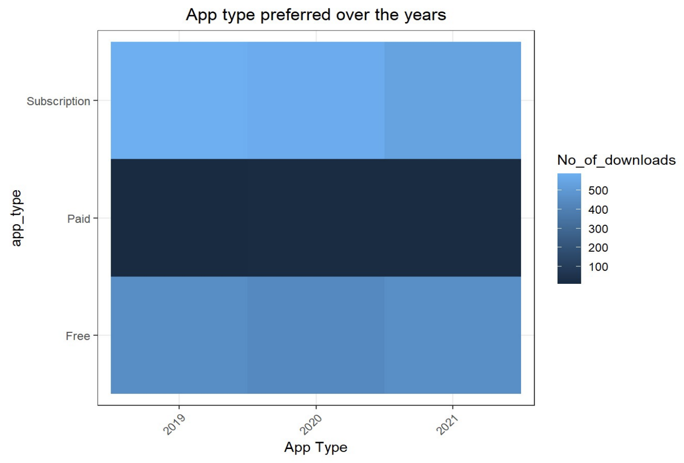
Plot 2:



- This bar chart shows the distribution of application downloads across different countries over the years.

- We can see that in the year 2019, India has the maximum number of downloads which reaches almost 300.
- It is also seen that there is no comparable differences in the downloads for the years 2020 and 2021 for all the countries.

Plot 3:



- This plot refers to a heat map that shows the application type (ie., whether it is free, paid, or subscribed) generally preferred over the years.
- We can observe that the maximum downloads occur mostly for the free and subscribed apps, subscribed ones are downloaded heavily and the most in the year 2019, and free apps are highly downloaded in the year 2019 and 2021.
- There are negligible downloads for the paid applications.

VI. Summary and Recommendation

We have derived desirable insights from our data that enabled us to understand how applications are being used by the users with good observations regarding the user purchase details and trends over the years on various app genres. We also determined how to analyze the use of applications based on the app ratings and from the count of downloads performed in each country.

As a recommendation, in the future, a web based interface can be constructed using FLASK making it more user-friendly to do querying and understand insights. Also, with the help of FLASK, CRUD (Create, Read, Update, Delete) operations become more efficient.
Moreover, using Flask- Forms library, we can incorporate constraint checking and form validation features making the entire implementation highly scalable and fault tolerant. The functionality of this DBMS can also be made efficient by including more complex relationships and entities as a result.