Design and implementation of a Cassandra database

Adrián Correa Cruz

Azul María García

José Luis Puente Bodoque

Nuria María Haba Díaz

December 27, 2022

Contents

1 Design steps

In order to design the Cassandra database, we have followed the steps below:

1. Draw the Entity-Relationship (ER) schema
2. Recognise the access patterns in the wording of the task: for each frequent query, draw a two-columns table with "given" and "find" as header
3. Implement the logical model: for each "given-find" table, define a new titled one that contains the list of fields along with its data type in any particular programming language
4. Implement the physical model: adjust the previous data types by Cassandra query language (CQL) data types and indicate what fields will become part of the primary key, clustering column(s) (also the sort direction) and the rest of column names that suits the query
5. Write the `CREATE TABLE` sentences in CQL

In this report we will focus on steps 1, 4 and 5 such as the task asks.

1.1 Entity-Relationship schema explanation

The first step consists of drawing the ER schema as a high-level conceptual data model. We have used draw.io for that purpose. The figure 1 shows the schema.

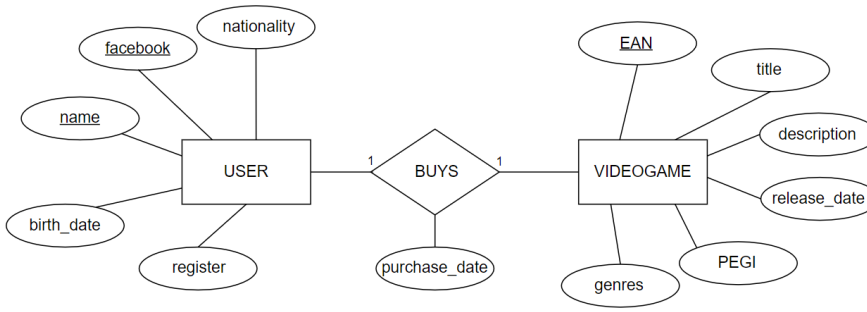


Figure 1: Entity-Relationship schema

The USER entity type has two key attributes, name and facebook, so that both identify uniquely in its own right each entity of the USER entity type. name key attribute stores the user's name and facebook key attribute stores an universal unique identifier (UUID). USER has also a nationality attribute, that stores the user's nationality, a birth_date attribute, that stores the user's birth date, and a register attribute that stores the user's register date in the database.

The VIDEOGAME entity type has the key attribute EAN which stores a EAN-13 (European Article Number of 13 digits) to identify a VIDEOGAME entity type uniquely. The title attribute is the title of the video game. description attribute is a large description of the game whereas is indicated the developer company, platform, recommended minimum age, the list of genres, if it is multiplayer or not and a score. release_date attribute stores the date when the video game started to be available in the market to buy, but our database also admits a pre-sale window of 30 days for some games. PEGI attribute stores the minimum recommended age to play a video game. genres attribute stores a set of genres which the video game is classified in.

The BUYS binary relationship type allows only one USER entity type and one VIDEOGAME entity type to participate at the same time. Hence a user can buy one video game only and one video game be bought only at the same time. purchase_date attribute records the purchase date.

Note: we will store dates in format "YYYY-MM-DD", according to the IS0 8601 standard.

1.2 Physical Model

On the following figures we are referring to the table on the right, but we will show the "given-find" tables they proceed from too.

So the figure 2 shows the users_by_name table in CQL data types. Given the user's name, we can find the complete user's information. We define name as the primary key in order to query by the user's name. The partition key is equivalent to the primary key. No clustering columns.

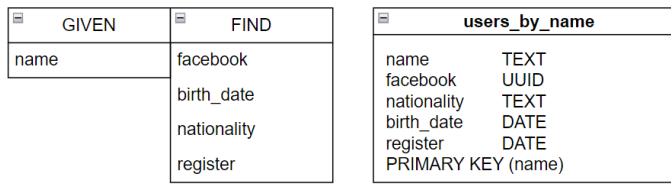


Figure 2: "Users by name" table

The figure 3 shows the users_by_facebook table in CQL data types. Given the user's facebook, we can find the complete user's information. We define facebook as the primary key in order to query by the user's facebook. The partition key is equivalent to the primary key. No clustering columns.

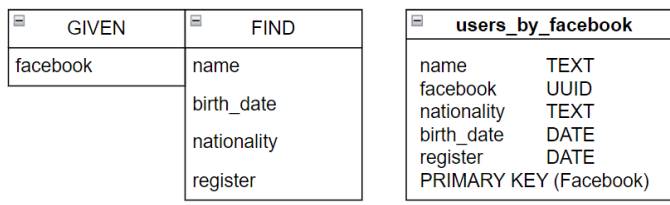


Figure 3: "Users by facebook" table

The figure 4 shows the titles_by_user table in CQL data types. Given the user's name, we can find the game titles and purchase dates. We define name as the primary key in order to query by the user's name. purchase_date is the clustering column that orders the table rows in a decreasing way.

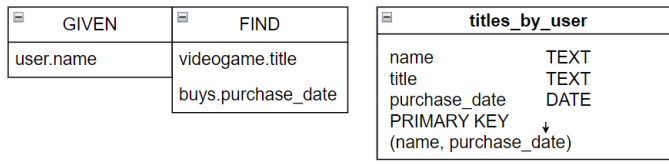


Figure 4: "Titles by user" CQL table

The figure 5 shows videogames table in CQL data types. The initial access pattern that produces this table do not provide a given value, so we will find the video game's information by the title game and EAN. If we want to retrieve a specific title from of a range of video games whose title is repeated, we must also specify EAN in the WHERE clause. The wording of the task says anything about the uniqueness of the games titles in the database.

So we define title and EAN as the composite partition key. PEGI and release_date are the clustering columns that orders the table in an ascending way.

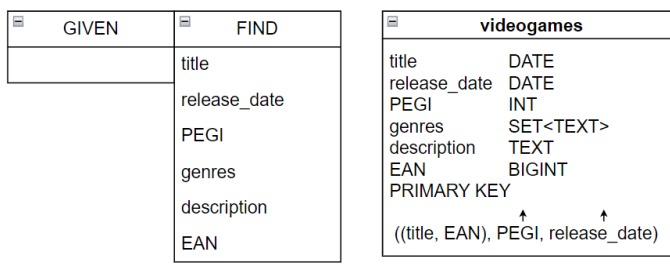


Figure 5: "Videogames" table

1.3 Create the CQL tables

For the users_by_name table the `CREATE TABLE` code is:

```
CREATE TABLE users_by_name (
    name            TEXT,
    facebook        UUID,
    birth_date      DATE,
    nationality     TEXT,
    register        DATE,
    PRIMARY KEY (name)
);
```

For the users_by_facebook table the `CREATE TABLE` code is:

```
CREATE TABLE users_by_facebook (
    name            TEXT,
    facebook        UUID,
    birth_date      DATE,
    nationality     TEXT,
    register        DATE,
    PRIMARY KEY (facebook)
);
```

For the titles_by_user table the `CREATE TABLE` code is:

```
CREATE TABLE titles_by_user (
    name            TEXT,
    title           TEXT,
    purchase_date   DATE,
    PRIMARY KEY (name, purchase_date)
) WITH CLUSTERING ORDER BY (purchase_date DESC);
```

Note that we specify decreasing order by using the `WITH CLUSTERING ORDER BY` clause.

For the videogames table the `CREATE TABLE` code is:

```
CREATE TABLE videogames (
    title           TEXT,
    EAN             BIGINT,
    release_date    DATE,
    PEGI            INT,
    description     TEXT,
```

```
    genres          SET<TEXT>,
    PRIMARY KEY ((title, EAN), PEGI, release_date)
);
```

2 Implementation steps

In order to implement the database, we have followed the steps below:

1. For each CQL table, create a CSV file that contains data. To generate random and a good amount of data we have used Excel skills and web applications on the Internet
2. Write the `COPY` sentences in CQL to import the CSV files
3. Write `videogamesdb.cql` script to gather the `CREATE TABLE` and `COPY` sentences
4. Run the script on Cassandra shell to build the database in the most efficient way in terms of time processing and resources usage

2.1 Create CSV files

It's time to generate the CSV files. For that purpose, we have made first four Excel worksheets shown in figures 6–9. The table headers are composed of the partition key(s) followed by the clustering column(s) and the rest of column key(s), in that specific order.

To fill the tables with data, we have used: Excel functions, to concatenate cells and select randomly between a range of cells; and random data generators on the Internet, such as appdevtools and fossbytes, to generate UUIDs and the user's name in a massive and random way.

videogamesdb .XLSX

Archivo Editar Ver Insertar Formato Datos Herramientas Ayuda Última modificación hace 5 minutos

A1 | name

	A	B	C	D	E
1	**name**	**facebook**	**birth_date**	**nationality**	**register**
2	Lawrence Curry	08c4c75c-0a38-46fb-9960-4b4e81fd00f6	1991-05-23	New Zealand	2004-06-07
3	Elmo Key	651e82fc-664c-48f9-8b62-b99f476c2121	1959-05-05	Mexico	2004-09-24
4	Ulysses Lott	257bbd88-1576-43d1-9870-7b7ee1410cb8	1995-10-13	Colombia	2009-04-30
5	Xavier Dunn	17f7f07c-d323-433f-bd87-1855b811f1d9	2002-04-04	Ukraine	2021-06-18
6	Uta Hodge	3f955ce7-1df5-43aa-9f48-a7a381d9d8df	1963-05-13	Philippines	2019-01-02
7	Carly Owen	cf184a9f-b13d-402c-9c56-1783c064ab53	1994-08-13	Vietnam	2012-08-04
8	Boris Galloway	9acc9d9e-844e-4cb9-a029-886dbc5eb47c	1989-04-13	Netherlands	2022-10-10
9	Maite Chan	9ebd9f97-5839-4822-a087-900410d44925	1959-10-07	Netherlands	2009-06-15
10	Ezra Franklin	d78f05d7-ae8b-41fb-9c74-7dded86155ea	1965-02-11	United States	2008-12-28
11	Hanna Mcintyre	e8cadd76-894f-4d9e-a443-8e33ba01c68b	1982-09-08	Russian Federation	2017-01-09

Figure 6: Excel Worksheet 1 for “users by name” table

A1 | facebook

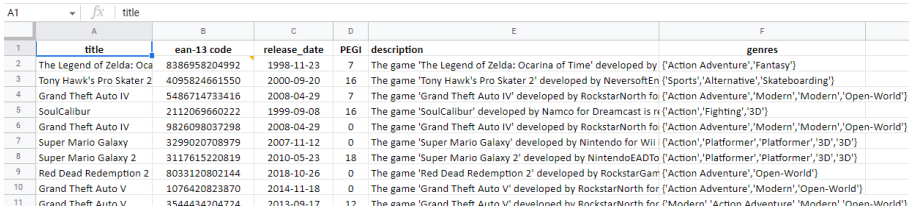
	A	B	C	D	E
1	**facebook**	**birth_date**	**name**	**nationality**	**register**
2	08c4c75c-0a38-46fb-9960-4b4e81fd00f6	1991-05-23	Lawrence Curry	New Zealand	2004-06-07
3	651e82fc-664c-48f9-8b62-b99f476c2121	1959-05-05	Elmo Key	Mexico	2004-09-24
4	257bbd88-1576-43d1-9870-7b7ee1410cb8	1995-10-13	Ulysses Lott	Colombia	2009-04-30
5	17f7f07c-d323-433f-bd87-1855b811f1d9	2002-04-04	Xavier Dunn	Ukraine	2021-06-18
6	3f955ce7-1df5-43aa-9f48-a7a381d9d8df	1963-05-13	Uta Hodge	Philippines	2019-01-02
7	cf184a9f-b13d-402c-9c56-1783c064ab53	1994-08-13	Carly Owen	Vietnam	2012-08-04
8	9acc9d9e-844e-4cb9-a029-886dbc5eb47c	1989-04-13	Boris Galloway	Netherlands	2022-10-10
9	9ebd9f97-5839-4822-a087-900410d44925	1959-10-07	Maite Chan	Netherlands	2009-06-15
10	d78f05d7-ae8b-41fb-9c74-7dded86155ea	1965-02-11	Ezra Franklin	United States	2008-12-28
11	e8cadd76-894f-4d9e-a443-8e33ba01c68b	1982-09-08	Hanna Mcintyre	Russian Federation	2017-01-09

Figure 7: Excel Worksheet 2 for “users by facebook” table

A1 | name

	A	B	C
1	**name**	**title**	**purchase_date**
2	Adrienne Burnett	City Crisis	2003-06-26
3	Kylynn Knight	Age of Empires	2003-08-11
4	Olivia Love	MLB Slugfest 20-03	2003-09-22
5	Amy Harris	Earthworm Jim	2003-10-21
6	Nita Abbott	Spy Hunter 2	2003-12-13
7	Petra Glass	Duck Dodgers Starring Daffy Duck	2004-01-17
8	Jelani Carter	Black & Bruised	2004-05-07
9	Adrienne Burnett	Vexx	2004-06-02
10	Graham Mosley	Super Bust-A-Move	2004-06-13
11	Jelani Carter	Homeworld	2004-06-17

Figure 8: Excel Worksheet 3 for “titles by user” table



	title	ean-13 code	release_date	PEGI	description	genres
2	The Legend of Zelda: Oca	8386958204992	1998-11-23	7	The game 'The Legend of Zelda: Ocarina of Time' developed by	{'Action Adventure','Fantasy'}
3	Tony Hawk's Pro Skater 2	4095824661550	2000-09-20	16	The game 'Tony Hawk's Pro Skater 2' developed by NeversoftEn	{'Sports','Alternative','Skateboarding'}
4	Grand Theft Auto IV	5486714733416	2008-04-29	7	The game 'Grand Theft Auto IV' developed by RockstarNorth fo	{'Action Adventure','Modern','Modern','Open-World'}
5	SoulCalibur	2112069660222	1999-09-08	16	The game 'SoulCalibur' developed by Namco for Dreamcast is r	{'Action','Fighting','3D'}
6	Grand Theft Auto IV	9826098037298	2008-04-29	0	The game 'Grand Theft Auto IV' developed by RockstarNorth fo	{'Action Adventure','Modern','Modern','Open-World'}
7	Super Mario Galaxy	3299020708979	2007-11-12	0	The game 'Super Mario Galaxy' developed by Nintendo for Wii	{'Action','Platformer','Platformer','3D','3D'}
8	Super Mario Galaxy 2	3117615220819	2010-05-23	18	The game 'Super Mario Galaxy 2' developed by NintendoEADTo	{'Action','Platformer','Platformer','3D','3D'}
9	Red Dead Redemption 2	8033120802144	2018-10-26	0	The game 'Red Dead Redemption 2' developed by RockstarGam	{'Action Adventure','Open-World'}
10	Grand Theft Auto V	1076420823870	2014-11-18	0	The game 'Grand Theft Auto V' developed by RockstarNorth for	{'Action Adventure','Modern','Open-World'}
11	Grand Theft Auto V	3544434204724	2013-09-17	12	The game 'Grand Theft Auto V' developed by RockstarNorth for	{'Modern' 'Action Adventure' 'Modern' 'Open-World'}

Figure 9: Excel Worksheet 4 for "videogames" table

Once we have our tables filled, then we save them as CSV file.

Note: It seemed us a better idea to separate the CSV values by a semicolon (';') rather than by a comma to avoid potential conflicts with the comma of the set of genres when importing the data to the database.

2.2 Import CSV files

To import the `users_by_name.csv` file, the code is:

```
COPY users_by_name(name,facebook,birth_date,
    nationality,register)
FROM 'var/lib/cassandra/users_by_name.csv'
WITH DELIMITER = ';' AND HEADER = 'true';
```

To import the `users_by_facebook.csv` file, the code is:

```
COPY users_by_facebook (facebook,birth_date,name,
    nationality,register)
FROM 'var/lib/cassandra/users_by_facebook.csv'
WITH DELIMITER = ';' AND HEADER = 'true';
```

To import the `titles_by_user.csv` file, the code is:

```
COPY titles_by_user (name,title,purchase_date)
FROM 'var/lib/cassandra/titles_by_user.csv'
WITH DELIMITER = ';' AND HEADER = 'true';
```

To import the `videogames.csv` file, the code is:

```
COPY videogames (title,EAN,PEGI,release_date,
   description,genres)
FROM 'var/lib/cassandra/videogames.csv'
WITH DELIMITER = ';' AND HEADER = 'true';
```

Note that we have included the option `DELIMITER` in the four CQL sentences to set a semicolon as delimiter.

2.3 The CQL script

We have written the `videogamesdb.cql` file to gather the `CREATE TABLE` and `COPY` code in a single script. The content file is shown below.

```
1  --Create tables
2
3  CREATE TABLE users_by_name (
4      name           TEXT,
5      facebook       UUID,
6      birth_date     DATE,
7      nationality    TEXT,
8      register       DATE,
9      PRIMARY KEY (name)
10 );
11
12 CREATE TABLE users_by_facebook (
13     name           TEXT,
14     facebook       UUID,
15     birth_date     DATE,
16     nationality    TEXT,
17     register       DATE,
18     PRIMARY KEY(facebook)
19 );
20
21 CREATE TABLE titles_by_user (
22     name           TEXT,
23     title          TEXT,
24     purchase_date  DATE,
25     PRIMARY KEY (name, purchase_date)
26 ) WITH CLUSTERING ORDER BY (purchase_date DESC);
27
28 CREATE TABLE videogames (
```

```
    title           TEXT,
    EAN             BIGINT,
    release_date    DATE,
    PEGI            INT,
    genres          SET<TEXT>,
    description     TEXT,
    PRIMARY KEY ((title, EAN), PEGI, release_date)
);

--Import CSV (semicolon as delimiter):

COPY users_by_name(name,facebook,birth_date,
    nationality,register) FROM 'var/lib/cassandra/
    users_by_name.csv' WITH DELIMITER = ';' AND HEADER
    = 'true';
COPY users_by_facebook (facebook,birth_date,name,
    nationality,register) FROM 'var/lib/cassandra/
    users_by_facebook.csv' WITH DELIMITER = ';' AND
    HEADER = 'true';
COPY titles_by_user (name,title,purchase_date) FROM '
    var/lib/cassandra/titles_by_user.csv' WITH
    DELIMITER = ';' AND HEADER = 'true';
COPY videogames (title,EAN,PEGI,release_date,
    description,genres) FROM 'var/lib/cassandra/
    videogames.csv' WITH DELIMITER = ';' AND HEADER = '
    true';
```

2.4 Build the database

To build the database we first must start the "cassandra-master" container on Docker and then follow the steps below:

1. Open the docker server on a CMD window:

```
docker exec -it cassandra-master bash
```

2. Type `cqlsh` to open a Cassandra shell:

```
cqlsh
```

3. Create `videogamesdb` keyspace:

```
CREATE KEYSPACE videogamesdb
WITH replication = {
    'class': 'SimpleStrategy',
    'replication_factor' : 3
};
```

4. Switch to `videogamesdb` keyspace:

```
use videogamesdb;
```

5. Execute `videogamesdb.cql`, which must be located in the docker folder where Cassandra database is running:

```
source 'var/lib/cassandra/videogamesdb.cql';
```

The column families should have been loaded successfully.

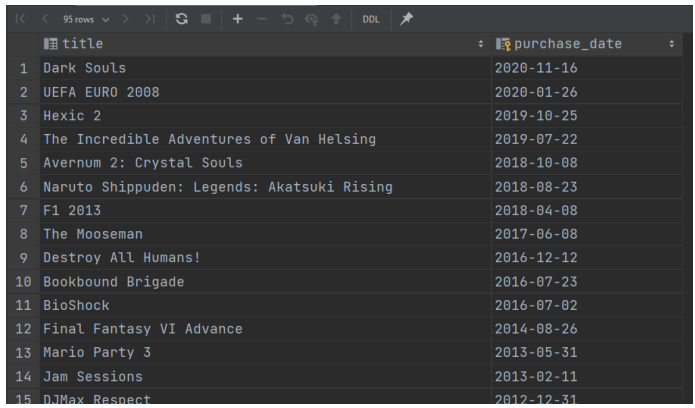
3 Queries

Query 1. Retrieve the purchase date and title of all video games bought by a given user and sorted decreasingly by purchase date.

For example, for "Ifeoma Bailey" the query request is:

```
SELECT title, purchase_date
FROM titles_by_user
WHERE name = 'Ifeoma Bailey'
ORDER BY purchase_date DESC
ALLOW FILTERING;
```

And the query result is (shows 15 of 95 results):



	title	purchase_date
1	Dark Souls	2020-11-16
2	UEFA EURO 2008	2020-01-26
3	Hexic 2	2019-10-25
4	The Incredible Adventures of Van Helsing	2019-07-22
5	Avernum 2: Crystal Souls	2018-10-08
6	Naruto Shippuden: Legends: Akatsuki Rising	2018-08-23
7	F1 2013	2018-04-08
8	The Mooseman	2017-06-08
9	Destroy All Humans!	2016-12-12
10	Bookbound Brigade	2016-07-23
11	BioShock	2016-07-02
12	Final Fantasy VI Advance	2014-08-26
13	Mario Party 3	2013-05-31
14	Jam Sessions	2013-02-11
15	DJMax Respect	2012-12-31

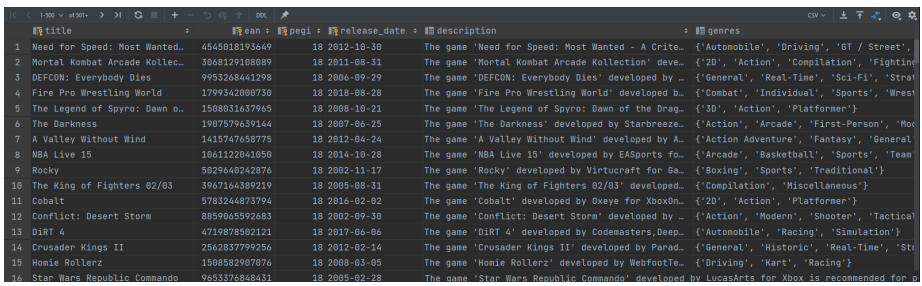
Figure 10: Query 1

Query 2. Retrieve data from video games whose minimum user age is greater or equal to 18 years old.

The query request is:

```
SELECT *
FROM videogames
WHERE pegi >= 18
ALLOW FILTERING;
```

And the query result is (shows 15 of 2924 results):



	title	ean	pegi	release_date	description	genres
1	Need for Speed: Most Wanted…	4545018193649	18	2012-10-30	The game 'Need for Speed: Most Wanted - A Crite…	{'Automobile', 'Driving', 'GT / Street',
2	Mortal Kombat Arcade Kollec…	3068129108889	18	2011-08-31	The game 'Mortal Kombat Arcade Kollection' deve…	{'2D', 'Action', 'Compilation', 'Fighting
3	DEFCON: Everybody Dies	9953268441298	18	2006-09-29	The game 'DEFCON: Everybody Dies' developed by …	{'General', 'Real-Time', 'Sci-Fi', 'Strat
4	Fire Pro Wrestling World	1799342080730	18	2018-08-28	The game 'Fire Pro Wrestling World' developed b…	{'Combat', 'Individual', 'Sports', 'Wrest
5	The Legend of Spyro: Dawn o…	1508031637965	18	2008-10-21	The game 'The Legend of Spyro: Dawn of the Drag…	{'3D', 'Action', 'Platformer'}
6	The Darkness	1907579639144	18	2007-06-25	The game 'The Darkness' developed by Starbreeze…	{'Action', 'Arcade', 'First-Person', 'Mod
7	A Valley Without Wind	1415747658775	18	2012-04-24	The game 'A Valley Without Wind' developed by A…	{'Action Adventure', 'Fantasy', 'General
8	NBA Live 15	1061122041050	18	2014-10-28	The game 'NBA Live 15' developed by EASports fo…	{'Arcade', 'Basketball', 'Sports', 'Team
9	Rocky	5029640242876	18	2002-11-17	The game 'Rocky' developed by Virtucraft for Ga…	{'Boxing', 'Sports', 'Traditional'}
10	The King of Fighters 02/03	3967164389219	18	2005-08-31	The game 'The King of Fighters 02/03' developed…	{'Compilation', 'Miscellaneous'}
11	Cobalt	5783244873794	18	2016-02-02	The game 'Cobalt' developed by Oxeye for XboxOn…	{'2D', 'Action', 'Platformer'}
12	Conflict: Desert Storm	8859065592683	18	2002-09-30	The game 'Conflict: Desert Storm' developed by …	{'Action', 'Modern', 'Shooter', 'Tactical
13	DiRT 4	4719878502121	18	2017-06-06	The game 'DiRT 4' developed by Codemasters,Deep…	{'Automobile', 'Racing', 'Simulation'}
14	Crusader Kings II	2562837799256	18	2012-02-14	The game 'Crusader Kings II' developed by Parad…	{'General', 'Historic', 'Real-Time', 'Str
15	Homie Rollerz	1508582907076	18	2008-03-05	The game 'Homie Rollerz' developed by WebfootTe…	{'Driving', 'Kart', 'Racing'}
16	Star Wars Republic Commando	9653376848431	18	2005-02-28	The game 'Star Wars Republic Commando' developed by LucasArts for Xbox is recommended for p

Figure 11: Query 2