INFO 6210

Data Management and Database Design

Physical Data Model and Normalization

Assignment 2

Report

Abstract

In this assignment, we used the data received from the API of thesneakerdatabase.com. We recollected the data from the API and use json to form the database and audited those data, delete all null one and then normalized the database into Third Norm Form. Besides, using we drew the UML model graph, conceptual model graph and physical model graph based on the interrelationship of entities and finally answered all given questions.

1.Data Sources

Web API

In this assignment, we choose to use the API called **Sneaker Database API.** This API is provided by thesneakerdatabase.com. It is one of the most popular websites that provides information for sneaker lovers with its well-covered sneakers data. The latest dataset we received has been stored as a Jason dataset, and will be used in further auditing process.

Here follows a screenshot of the dataset we got:


```
{'count': 25750,
 'results': [{'id': '87e9ea14-251f-42cb-b5a0-a190e8f15098',
   'brand': 'Vans',
   'colorway': 'Multi-Color',
   'gender': 'men',
   'media': {'imageUrl': 'https://stockx.imgix.net/Vans-Comfycush-Slip-On-Half-Big-Checker.png?fit=fill&bg=FFFFFF&w=700&h=500&auto=forma
t,compress&trim=color&q=90&dpr=2&updated_at=1578035967',
    'smallImageUrl': 'https://stockx.imgix.net/Vans-Comfycush-Slip-On-Half-Big-Checker.png?fit=fill&bg=FFFFFF&w=300&h=214&auto=format,co
mpress&trim=color&q=90&dpr=2&updated_at=1578035967',
    'thumbUrl': 'https://stockx.imgix.net/Vans-Comfycush-Slip-On-Half-Big-Checker.png?fit=fill&bg=FFFFFF&w=140&h=100&auto=format,compres
s&trim=color&q=90&dpr=2&updated_at=1578035967'},
   'releaseDate': '2020-12-31 23:59:59',
   'retailPrice': 65,
   'styleId': None,
   'title': 'Vans Comfycush Slip-On Half Big Checker',
   'year': 2020},
  {'id': '9595bf3d-d799-4084-ae3a-49e08360b2d5',
   'brand': 'Jordan',
   'colorway': 'Dark Mocha/Infrared 23-Black',
```


Figure 1-1-1 Screenshot of Data Gathered from Web API


```
  {'id': 'f78e239f-7600-463d-8d1a-a2f7b8c68c0f',
   'brand': 'Jordan',
   'colorway': 'White/Fire Red/Black',
   'gender': 'toddler',
   'media': {'imageUrl': 'https://stockx.imgix.net/Air-Jordan-5-Retro-Fire-Red-2020-TD.jpg?fit=fill&bg=FFFFFF&w=700&h=500&auto=format,co
mpress&trim=color&q=90&dpr=2&updated_at=1581619811',
    'smallImageUrl': 'https://stockx.imgix.net/Air-Jordan-5-Retro-Fire-Red-2020-TD.jpg?fit=fill&bg=FFFFFF&w=300&h=214&auto=format,compre
ss&trim=color&q=90&dpr=2&updated_at=1581619811',
    'thumbUrl': 'https://stockx.imgix.net/Air-Jordan-5-Retro-Fire-Red-2020-TD.jpg?fit=fill&bg=FFFFFF&w=140&h=100&auto=format,compress&tr
im=color&q=90&dpr=2&updated_at=1581619811'},
   'releaseDate': '2020-03-25 23:59:59',
   'retailPrice': 60,
   'styleId': '440890-102',
   'title': 'Jordan 5 Retro Fire Red 2020 (TD)',
   'year': 2020},
  {'id': '7bd3982c-f1f4-41f8-a884-8e14eacacba8',
   'brand': 'Jordan',
```


Figure 1-1-2 Screenshot of Data Gathered from Web API

Then we set up name for columns, reformed the json data into a more readable version.

	id	brand
0	87e9ea14-251f-42cb-b5a0-a190e8f15098	Vans
1	9595bf3d-d799-4084-ae3a-49e08360b2d5	Jordan
2	ee5c25ca-ac30-4de1-9d3f-65168b103361	Jordan
3	04b5280a-b04d-43eb-9ced-dc868a21779e	Jordan
4	52454854-3a4d-4db7-8c38-cca3705b3600	Jordan
5	c8535733-1136-49c1-9991-2fd2e41ac925	Jordan
6	e9b8aa8b-1a10-42a0-bec0-7bc2fd9cfb5f	Jordan
7	babbc835-5c05-4583-9408-655091277a13	Jordan
8	ca736d9e-c96e-40d6-9913-0f8e516680e6	Jordan
9	19486669-5518-4539-a2f8-881b8c56c44b	Nike
10	038a9659-c88a-4cce-92cf-687e29856e2b	Jordan
11	f7a447c7-27f5-4b92-acb2-bd60580b8267	Jordan
12	fc5c20cd-4acc-4d60-8154-40c9ca2cc477	Jordan
13	a1cb8abf-13fd-4573-9ee0-f4e20435942c	Jordan
14	033454e1-af0e-4a30-9af1-5b69d2d8649b	Jordan
15	7bf95a86-42af-4e00-b3ba-b9978ed0c7cf	Jordan
16	7deca14b-1bef-4c11-9005-ddb2f70faf17	Jordan

Figure 1-1-3 Screenshot of Data Gathered from Web API

	colorway	gender
0	Multi-Color	men
1	Dark Mocha/Infrared 23-Black	men
2	Apple Green/Black-Yellow Strike-Black	men
3	Stone Blue/Legend Blue-Obsidian	toddler
4	Stone Blue/Legend Blue-Obsidian	child
5	Stone Blue/Legend Blue-Obsidian	men
6	Stone Blue/Legend Blue-Obsidian	preschool
7	White/Pine Green-Neutral Grey-Muslin	men
8	Black/Fire Red-Cement Grey-White	men
9	Black/Electric Green	men
10	Black/Fire Red-Grape Ice-New Emerald	men
11	Black/Fire Red-Grape Ice-New Emerald	preschool

Figure 1-1-4 Screenshot of Data Gathered from Web API

	releaseDate	retailPrice	styleId
0	2020-12-31 23:59:59	65.0	None
1	2020-09-26 23:59:59	190.0	CT8011-200
2	2020-09-12 23:59:59	225.0	CK6631-307
3	2020-08-08 23:59:59	60.0	None
4	2020-08-08 23:59:59	140.0	None
5	2020-08-08 23:59:59	190.0	130690-404
6	2020-08-08 23:59:59	80.0	None
7	2020-08-05 23:59:59	225.0	CK6630-100
8	2020-07-09 23:59:59	170.0	919712-006
9	2020-07-01 23:59:59	NaN	None
10	2020-05-16 23:59:59	200.0	CZ1786-001
11	2020-05-16 23:59:59	90.0	None
12	2020-05-16 23:59:59	65.0	None
13	2020-05-09 23:59:59	170.0	555088-041

Figure 1-1-5 Screenshot of Data Gathered from Web API

2.Data Auditing

Audit validity/ accuracy

In this part, we will audit the validity of our data. The main process is as below:

1.Check whether there are any null or duplicates

2.Delete the null or duplicated data

In the first version of the rawSneakerJsonData table, using *.isnull* command, we spotted several null values. Among those 49 rows of data.

```
id            False
brand         False
colorway      False
gender        False
releaseDate   False
retailPrice    True
styleId        True
title         False
year          False
dtype: bool
```

Figure 2-1-1 screenshot of Data validity auditing process and result

We then cleaned all those data with any null values and at last, 25 rows of data remained. Because currently there are no duplicates and null values, the database is already cleaned.

```
id            False
brand         False
colorway      False
gender        False
releaseDate   False
retailPrice   False
styleId       False
title         False
year          False
dtype: bool
```

Figure 2-1-2 screenshot of Data validity auditing process and result

And here is the screenshot of those remaining data.

15	Neutral Grey/White-True Red-Black	men
19	Court Purple/White-Black	men
24	White/Black-Metallic Silver-Fire Red	men
26	White/Fire Red/Black	preschool
27	White/Fire Red/Black	toddler
29	Cool Grey/Volt-Wolf Grey-Anthracite	child
30	Black/Orange-Cement Grey-Outdoor green	men
32	White/Valor Blue-Tech Grey	men
33	University Red/Black-Court Purple	men
34	Gym Red/Gym Red-White	men
35	Gym Red/Gym Red-White	child
36	Football Grey/Saffron Quartz-Fire Pink-Sail	women
37	Dark Pony/Burgundy Ash-Desert Dust	women
38	Soar/Total Orange-Volt-Platinum Tint	men
39	Black/Black-Parachute Beige-Petra Brown	men
41	Orange/Orange/Orange	men
44	Tail Light/Tail Light/Tail Light	men
45	Earth/Earth/Earth	men

Figure 2-1-3 screenshot of Data validity auditing process and result

Audit Completeness

There are 7 columns contained in the dataset, which are id, brand, colorway, gender, releasedate, retailPrice, styleId, title, year. These data cover all the important entities of sneakers which should

have, and thus satisfy the completeness of database.

Audit Consistency and Uniformity

The datasets are inner related by a common entity, title, which relates all the datasets. Thus, the final combined database is considered uniformed and consistent.

3.Reformatting the database

In this part, we used .loc command to reformate the database for the following normalizing process. After the reformatting process, the data has been divided and filled several different tables.

The Color_gender table contains columns of 'id','colorway','gender', the Release_time table contains columns of 'id','releaseDate','year', whereas the Brand_Retailprice table is consisted of columns named 'id','title','brand' and 'retailPrice'. Here follows the screenshot of results of the reformatting process.

	id	colorway	gender
1	9595bf3d-d799-4084-ae3a-49e08360b2d5	Dark Mocha/Infrared 23-Black	men
2	ee5c25ca-ac30-4de1-9d3f-65168b103361	Apple Green/Black-Yellow Strike-Black	men
5	c8535733-1136-49c1-9991-2fd2e41ac925	Stone Blue/Legend Blue-Obsidian	men
7	babbc835-5c05-4583-9408-655091277a13	White/Pine Green-Neutral Grey-Muslin	men
8	ca736d9e-c96e-40d6-9913-0f8e516680e6	Black/Fire Red-Cement Grey-White	men
10	038a9659-c88a-4cce-92cf-687e29856e2b	Black/Fire Red-Grape Ice-New Emerald	men
13	a1cb8abf-13fd-4573-9ee0-f4e20435942c	Black/White-Game Royal-Black	men
15	7bf95a86-42af-4e00-b3ba-b9978ed0c7cf	Neutral Grey/White-True Red-Black	men
19	32c8529c-2b39-44e3-8b6d-7b99ec546380	Court Purple/White-Black	men
24	d7add4ba-84dd-4639-977f-661ce1959552	White/Black-Metallic Silver-Fire Red	men
26	c0591fd8-e359-4028-88dd-94b20165ab5f	White/Fire Red/Black	preschool
27	f78e239f-7600-463d-8d1a-a2f7b8c68c0f	White/Fire Red/Black	toddler
29	e227866b-ddab-4ea9-b54e-11e07161e25c	Cool Grey/Volt-Wolf Grey-Anthracite	child
30	ba80c366-c551-449e-9da4-75d7ce26c49b	Black/Orange-Cement Grey-Outdoor green	men
32	652d8004-d441-466d-a725-21d48600f624	White/Valor Blue-Tech Grey	men
33	edfa8bad-5097-4da9-a4c8-b6e3c2dbc1b0	University Red/Black-Court Purple	men

Figure 3-1-1 screenshot of database reformatting process and result

	id	releaseDate	year
1	9595bf3d-d799-4084-ae3a-49e08360b2d5	2020-09-26 23:59:59	2020
2	ee5c25ca-ac30-4de1-9d3f-65168b103361	2020-09-12 23:59:59	2020
5	c8535733-1136-49c1-9991-2fd2e41ac925	2020-08-08 23:59:59	2020
7	babbc835-5c05-4583-9408-655091277a13	2020-08-05 23:59:59	2020
8	ca736d9e-c96e-40d6-9913-0f8e516680e6	2020-07-09 23:59:59	2020
10	038a9659-c88a-4cce-92cf-687e29856e2b	2020-05-16 23:59:59	2020
13	a1cb8abf-13fd-4573-9ee0-f4e20435942c	2020-05-09 23:59:59	2020
15	7bf95a86-42af-4e00-b3ba-b9978ed0c7cf	2020-04-08 23:59:59	2020
19	32c8529c-2b39-44e3-8b6d-7b99ec546380	2020-04-04 23:59:59	2020
24	d7add4ba-84dd-4639-977f-661ce1959552	2020-03-28 23:59:59	2020
26	c0591fd8-e359-4028-88dd-94b20165ab5f	2020-03-25 23:59:59	2020
27	f78e239f-7600-463d-8d1a-a2f7b8c68c0f	2020-03-25 23:59:59	2020
29	e227866b-ddab-4ea9-b54e-11e07161e25c	2020-03-21 23:59:59	2020
30	ba80c366-c551-449e-9da4-75d7ce26c49b	2020-03-14 23:59:59	2020
32	652d8004-d441-466d-a725-21d48600f624	2020-03-07 23:59:59	2020
33	edfa8bad-5097-4da9-a4c8-b6e3c2dbc1b0	2020-03-05 23:59:59	2020
34	54a24b2d-4353-4881-9a03-75126bdb9fa6	2020-03-01 23:59:59	2020
35	ae674014-8dbf-4851-86bd-9cd91ada6934	2020-03-01 23:59:59	2020
36	4ed9417c-ac31-4ebe-9d7a-b6d6f06c7453	2020-03-01 23:59:59	2020
37	b0c458a8-0932-438f-ae00-f366f0547035	2020-03-01 23:59:59	2020

Figure 3-1-2 screenshot of database reformatting process and result

	id	title	brand	retailPrice
1	9595bf3d-d799-4084-ae3a-49e08360b2d5	Jordan 10 Retro Dark Mocha	Jordan	190.0
2	ee5c25ca-ac30-4de1-9d3f-65168b103361	Jordan 5 Retro SE Oregon	Jordan	225.0
5	c8535733-1136-49c1-9991-2fd2e41ac925	Jordan 12 Retro Stone Blue	Jordan	190.0
7	babbc835-5c05-4583-9408-655091277a13	Jordan 4 Retro Pine Green	Jordan	225.0
8	ca736d9e-c96e-40d6-9913-0f8e516680e6	Jordan 11 Retro Low IE Black Cement	Jordan	170.0
10	038a9659-c88a-4cce-92cf-687e29856e2b	Jordan 5 Retro Top 3	Jordan	200.0
13	a1cb8abf-13fd-4573-9ee0-f4e20435942c	Jordan 1 Retro High Black Game Royal	Jordan	170.0
15	7bf95a86-42af-4e00-b3ba-b9978ed0c7cf	Jordan 6 Retro Hare	Jordan	190.0
19	32c8529c-2b39-44e3-8b6d-7b99ec546380	Jordan 1 Retro High Court Purple White	Jordan	160.0
24	d7add4ba-84dd-4639-977f-661ce1959552	Jordan 5 Retro Fire Red Silver Tongue (2020)	Jordan	200.0
26	c0591fd8-e359-4028-88dd-94b20165ab5f	Jordan 5 Retro Fire Red 2020 (PS)	Jordan	80.0
27	f78e239f-7600-463d-8d1a-a2f7b8c68c0f	Jordan 5 Retro Fire Red 2020 (TD)	Jordan	60.0
29	e227866b-ddab-4ea9-b54e-11e07161e25c	Jordan 4 Retro SE Neon (GS)	Jordan	160.0
30	ba80c366-c551-449e-9da4-75d7ce26c49b	Nike SB Dunk Low Safari	Nike	150.0
32	652d8004-d441-466d-a725-21d48600f624	Jordan 3 Retro UNC (2020)	Jordan	190.0
33	edfa8bad-5097-4da9-a4c8-b6e3c2dbc1b0	Jordan 2 Retro Raptors	Jordan	190.0
34	54a24b2d-4353-4881-9a03-75126bdb9fa6	Jordan 1 Low Gym Red White	Jordan	90.0
35	ae674014-8dbf-4851-86bd-9cd91ada6934	Jordan 1 Low Gym Red White (GS)	Jordan	75.0
36	4ed9417c-ac31-4ebe-9d7a-b6d6f06c7453	Air Max 95 Football Grey Saffron Quartz Fire P...	Nike	170.0
37	b0c458a8-0932-438f-ae00-f366f0547035	Air Force 1 Shell Burgundy Ash (W)	Nike	150.0
38	8f8c3f55-0996-4b8d-b89c-4c4a14ec5bfd	Air Max 270 React Eng Soar Total Orange Volt	Nike	150.0
39	8707af69-9d2b-47f4-bd53-b98501db1592	Nike SB Dunk Low Travis Scott	Nike	150.0
41	55d7e9f2-501a-4aae-b6c2-140f5b238d11	adidas Yeezy Boost 700 MNVN Orange	adidas	220.0

Figure 3-1-3 screenshot of database reformatting process and result

4.Normalization Analysis

For our sneaker database, we generally have three database: 1.Color_Gender table, 2.Release_time table, 3.Brand_Retailprice table.

Since the id of the sneaker data is the primary key of each table and is already unique. There are no multi-values in each column, in other words, the value in each column is atomic. Also, no

two columns store same type of values. Thus, the above database is already in First-normal-form.

However, the release year is dependent on the release data. So, the database is not in Third norm. Similarly, the title attribute in the Brand_Retail price is dependent on the id, and the brand and retailPrice attribute is dependent on the title. Thus, the Brand_Retail price also has transitive dependency.

Thus, we would separate the Release_time table and Brand_RetailPrice table to satisfy the database into second normal form and third normal form.

After the normalization, the sneaker database is now in Third Norm Form. The sneaker database consists of 5 tables, which are:

1.Sneaker_Retailprice table(primary key is title)

2.Sneaker_Title table(primary key is id, foreign key is title)

3.Release_Date table(Primary key is id, foreign key is releaseDate)

4.Release_Year table(Primary key is releaseDate)

5.Color_Gender table(Primary key is id)

Here follows the screenshot of results of the normalization process.

	id	title
1	9595bf3d-d799-4084-ae3a-49e08360b2d5	Jordan 10 Retro Dark Mocha
2	ee5c25ca-ac30-4de1-9d3f-65168b103361	Jordan 5 Retro SE Oregon
5	c8535733-1136-49c1-9991-2fd2e41ac925	Jordan 12 Retro Stone Blue
7	babbc835-5c05-4583-9408-655091277a13	Jordan 4 Retro Pine Green
8	ca736d9e-c96e-40d6-9913-0f8e516680e6	Jordan 11 Retro Low IE Black Cement
10	038a9659-c88a-4cce-92cf-687e29856e2b	Jordan 5 Retro Top 3
13	a1cb8abf-13fd-4573-9ee0-f4e20435942c	Jordan 1 Retro High Black Game Royal
15	7bf95a86-42af-4e00-b3ba-b9978ed0c7cf	Jordan 6 Retro Hare
19	32c8529c-2b39-44e3-8b6d-7b99ec546380	Jordan 1 Retro High Court Purple White
24	d7add4ba-84dd-4639-977f-661ce1959552	Jordan 5 Retro Fire Red Silver Tongue (2020)
26	c0591fd8-e359-4028-88dd-94b20165ab5f	Jordan 5 Retro Fire Red 2020 (PS)
27	f78e239f-7600-463d-8d1a-a2f7b8c68c0f	Jordan 5 Retro Fire Red 2020 (TD)
29	e227866b-ddab-4ea9-b54e-11e07161e25c	Jordan 4 Retro SE Neon (GS)
30	ba80c366-c551-449e-9da4-75d7ce26c49b	Nike SB Dunk Low Safari
32	652d8004-d441-466d-a725-21d48600f624	Jordan 3 Retro UNC (2020)
33	edfa8bad-5097-4da9-a4c8-b6e3c2dbc1b0	Jordan 2 Retro Raptors
34	54a24b2d-4353-4881-9a03-75126bdb9fa6	Jordan 1 Low Gym Red White
35	ae674014-8dbf-4851-86bd-9cd91ada6934	Jordan 1 Low Gym Red White (GS)
36	4ed9417c-ac31-4ebe-9d7a-b6d6f06c7453	Air Max 95 Football Grey Saffron Quartz Fire P...
37	b0c458a8-0932-438f-ae00-f366f0547035	Air Force 1 Shell Burgundy Ash (W)
38	8f8c3f55-0996-4b8d-b89c-4c4a14ec5bfd	Air Max 270 React Eng Soar Total Orange Volt

Figure 4-1-1 screenshot of database normalization process and result

5. Graph

5.1 UML model graph

From the UML model graph below we could see that the Release_Year 'IS A' Release_Date. Release_Year inherited from the Release_Date, and thus is a Release_Date. For a single Sneaker, it has the attributes of Color_Gender, Release_Date and Sneaker_Retailprice. Thus, the Sneaker_Title 'Has A' Sneaker_Retailprice, Release_Date and Color_gender.

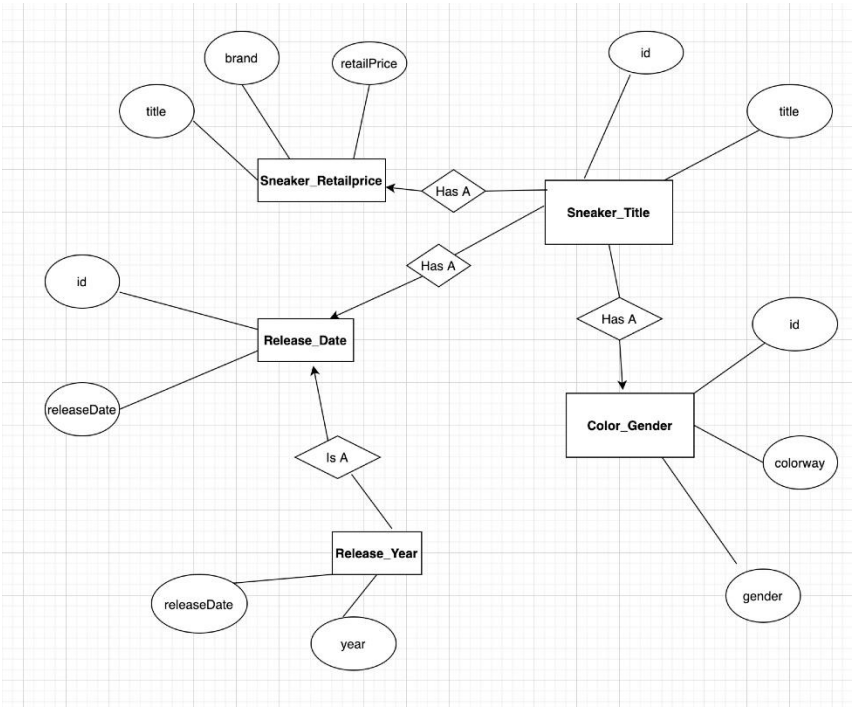

Figure 5-1-1 UML model graph

5.2 Conceptual Data Modeling

From the graph below we could see that in the sneaker database, there are 5 tables, namely, Sneaker_Retailprice, Sneaker_Title, Release_Date, Release_Year, Color_Gender. The Release_Date and Sneaker_title has a Many-to-Many relationship. The Color_Gender and Sneaker_Title has a Many-to-Many relationship. The Release_Date table and Release_Year table has a One-to-Many relationship, since in one year there could be many detailed release date.The Sneaker_Retailprice and Sneaker_title is also One-to-Many relationship, since one sneaker could only have one retail price, but multiple sneakers could be sold at one price.

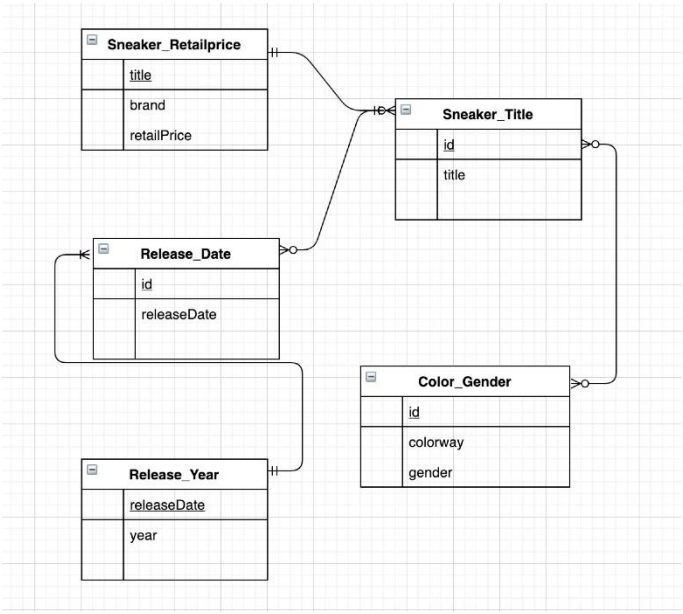

Figure 5-2-1 Conceptual Data Modeling Graph

5.3 Physical Model

In the following Physical model graph, we can see the detailed primary keys, foreign keys, table names, column names and data type of the database. The foreign key of Sneaker_Title title is the primary key of the Sneaker_Retailprice. The foreign key releaseDate is the primary key of table Release_Year. The primary key of table Sneaker_Title, Release_Date and Color_Gender are the same id.

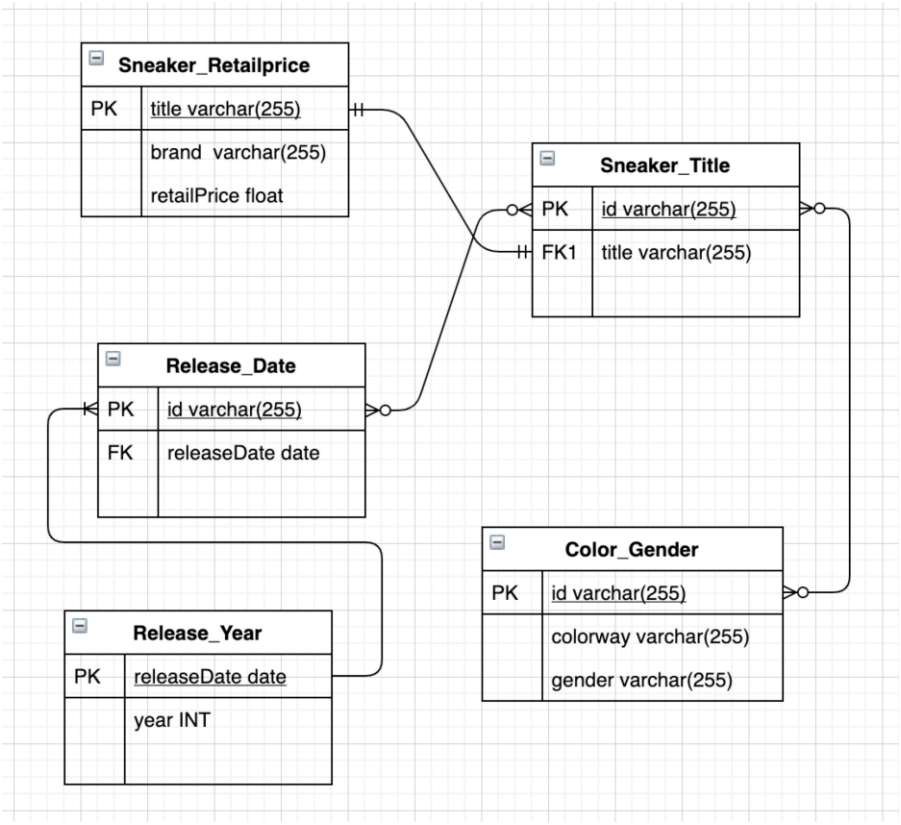

Figure 5-3-1 Physical Model Graph

6 Answering Questions

Question 1:

What are the ranges, data types and format of all of the attributes in your entities?

Answers: Entity1: Sneaker_Title Attritutes: id varchar(255) title varchar (255)
Entity2: Sneaker_Retailprice Attritutes: title varchar(255) brand varchar (255) retailPrice Float
Entity3: Release_Date Attritutes: Id varchar(255) releaseDate date
Entity4: Color_Gender Attritutes: colorway varchar(255) gender varchar(255)
Entity5: Release_Year Attritutes: releaseDate date year INT

Question2:

When should you use an entity versus attribute? (Example: address of a person could be modeled as either)

Answers:

The original meaning of one entity is a subject, which refers to a table in the database. This

entity must be distinguishable from other entities. Entities could be abstract or specific. The attribute is owned by entities and are used to describe each entity. The attribute could be described as a function that maps the entity that it belongs to the corresponding database.

While we first form a database, we need to establish the relationship between entities. When we dive into the entities, we need to be specific on each attribute, and thus, we need to add value to attributes, datatype and ranges to attributes.

The address of a person could indeed be modeled either as an attribute and an entity. If we define the address as an entity, the attribute belongs to it could be city, country, zip code, etc. If the address is defined as a person, the entity it belongs to could be a person. The person entity could have entities like name address, sex, age, etc.

Question3:

When should you use an entity or relationship, and placement of attributes? (Example: a manager could be modeled as either)

A manager could be modeled as either. If a manager is modeled as an entity, a manager is a one category in the database, in other word, a table in the database. In this manager entity, attributes could be manageer_id, name, age, sex, phone number, etc. If a manager is modeled as an attribute, it must belong to a certain entity, and manager is one of the attributes of this entity. This time, the entity that manager belongs to could be a department in a company, could be a job. The other attributes in the entity Manager belongs to could be normal staff, CEOs, etc. If a manager is modeled as a relationship, manager should be the connection of two entities. It is responsible for establishing relation between two entities. For example, there could be a managing relationship table describing the managing hierarchy of the company, like certain employees are under the leadership of certain managers.

Question4:

How did you choose your keys? Which are unique?

In my database, I used id of the sneaker as the primary key for Sneaker_Title, Release_Date and Color_Gender table. Since the Id of each sneaker is the unique identifier of all these three tables, Id could be the primary key of all these three tables. When normalizing the database, we seperated the Sneaker_Retailprice from the Sneaker_Title table, the primary key of the new table is the title of the sneaker, which is also unique. We also separated the Release_Year from the Release_data database. The primary key of the Release_year table is the releaseDate.

Question5:

Did you model hierarchies using the “ISA” design element? Why or why not

ISA basically represents the inheritance relationship of different entities. In our UML graph model, we used ISA relationship to represent the relationship of Release_Date table and Release_Year table. Basically, Release_Year is another form of Release_Date, just being more general. This represents the inheritance relationship, and thus, is an ISA relationship.

Question6:

Were there design alternatives? What are their tradeoffs: entity vs. attribute, entity vs. relationship, binary vs. ternary relationships?

There are indeed alternatives in my database design.When forming the Color_Gender table, we first decided to combine the Color_Gender table into the Sneaker_Title table. But it could be a little reduntant for the sneaker_title database, since the entity of Colorway and gender does not have a direct strong relationship with the Sneaker_Title. Thus, separting the Color_Gender table from the Sneaker_Title database would make more sense. Since the Sneaker_Title database is connected to Color_Gender table, Release_Date table and Sneaker_Retailprice table, it forms a ternary relationship.

Question7:

Where are you going find real-world data populate your model?

Currently all of my database data are from API:
https://api.thesneakerdatabase.com/v1/sneakers?limit=50&page=.

Questions you must answer about your physical model:

Are all the tables in 1NF?

The original data is already in 1NF, since the primary key could uniquely identify a record, the values in each column of a table are atomic and there are no repeating groups.

Are all the tables in 2NF?

The original data is already in 2NF, since there are no composite keys, and thus, no partial dependencies. I would argue that the Release_Year are not calculated from the Release_Date, since they have different data type, and thus has nothing to do with calculation(the Release_Date is string and Release_Year is INT).

Are all the tables in 3NF?

The original data is not in 3NF, since the Release_Year is dependent on the Release_Date, and Release_Date is dependent on the Id of the sneaker. Also, the Retail_Price and Brand is dependent on the title of the sneaker, and the title is dependent on the Id. So, there are transitive dependencies in original Release_Time table and Sneaker_title table. So, we separated the above two tables to normalize the database into 3NF.

Final Report

In this assignment, the following files are generated and used: sneaker.csv, gender.csv, Sneaker_title.csv, colorway.csv, retailPrice.csv,test1.csv. These data are gathered from three different source, web API, web scraping and raw data, which are then merged to form the final conceptual model.

Conclusion

In this assignment, firstly, we gathered data from different source, which are web API, web scraper and raw data. Then the data is cleaned, reformatted and combined. During the process, the null values are cleaned, and the structure and relationship of the data is much clearer. Finally, we formed the conceptual model and draw the Entity Relationship graph to further clarify the model.

Contribution

Original contribution: 40% By External source: 20% Provided by the TA documents : 40%.

Citations

https://app.swaggerhub.com/apis-docs/tg4solutions/the-sneaker-database/1.0.0#/sneakers/getSneakers

http://www.thesneakerdatabase.com/

https://towardsdatascience.com/web-scraping-using-selenium-and-beautifulsoup-99195cd70a58

http://unclechen.github.io/2016/12/11/python%E5%88%A9%E7%94%A8beautifulsoup+selenium%E8%87%AA%E5%8A%A8%E7%BF%BB%E9%A1%B5%E6%8A%93%E5%8F%96%E7%BD%91%E9%A1%B5%E5%86%85%E5%AE%B9/

License