**App Store Database Model**

**Created by**

**Nerkar Nilesh**

**Nuid# 001899192**

Version 3.0



**INFO 6210 34339**

**Data Management and Database Design**

**Spring 18**

**Table of Contents**

[**Introduction** 3](#_Toc510813350)

[**Description of Clusters** 4](#_Toc510813351)

[**Applications Cluster** 5](#_Toc510813352)

[**Applications** 5](#_Toc510813353)

[**Applications\_Archive** 8](#_Toc510813354)

[**Application\_Reviews** 8](#_Toc510813355)

[**Reviews\_Archive** 9](#_Toc510813356)

[**Application\_Compatibility** 10](#_Toc510813357)

[**Categories** 11](#_Toc510813358)

[**Application\_Documents** 12](#_Toc510813359)

[**Country\_Languages** 13](#_Toc510813360)

[**Downloads Cluster** 14](#_Toc510813361)

[**Downloads** 14](#_Toc510813362)

[**Devices** 16](#_Toc510813363)

[**Device\_Types** 17](#_Toc510813364)

[**Users Cluster** 18](#_Toc510813365)

[**Users** 18](#_Toc510813366)

[**User\_Account\_Details** 20](#_Toc510813367)

[**Purchases** 21](#_Toc510813368)

[**Credit\_Card\_Details** 23](#_Toc510813369)

[**Gift\_Cards** 24](#_Toc510813370)

[**User Addresses** 25](#_Toc510813371)

[**Developers Cluster** 27](#_Toc510813372)

[**Developers** 27](#_Toc510813373)

[**Payments** 28](#_Toc510813374)

[**Memberships** 30](#_Toc510813375)

[**Bank\_Details** 31](#_Toc510813376)

[**Developer Addresses** 32](#_Toc510813377)

[**Systems Cluster** 33](#_Toc510813378)

[**Application\_Files** 33](#_Toc510813379)

[**Relationship Description** 35](#_Toc510813380)

[**Related Documents** 37](#_Toc510813381)

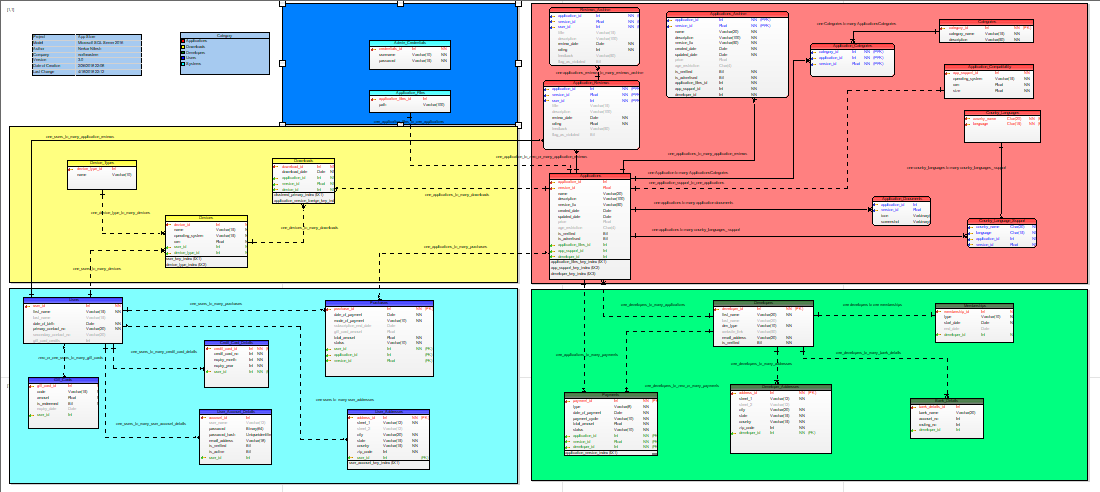
[**Revision History** 37](#_Toc510813382)

# **Introduction**

The primary idea behind the model is getting to know how an application is downloaded/purchased by a user and how the developers can get benefits for their work. Furthermore, the users are also able to download the latest versions of the application. The user’s downloads will be tracked with respect to its devices as well as the version number. We will also get to know the different types of applications which are under one categories.

Although the app store database is created and maintained in NO-SQL database, here we have implemented the same in relational database.

*App store data model*

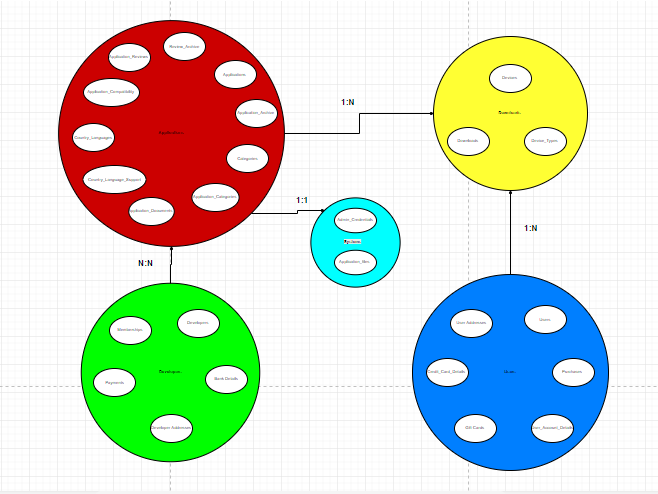


(Kindly find attached image in the documents section for bigger picture)

# **Description of Clusters**

A cluster is collection of a similar types of entities which is linked to a common behavior. Here, we have five clusters in the app store model namely Developers, Applications, Downloads, Users and Systems. A user downloads an application which is created by the developer. A Developer creates many Applications and an Application is Downloaded many times. Similarly, a User has many Downloads. An Application has a System cluster linked to it where the meta data is stored of the applications.

*Cluster Diagram*

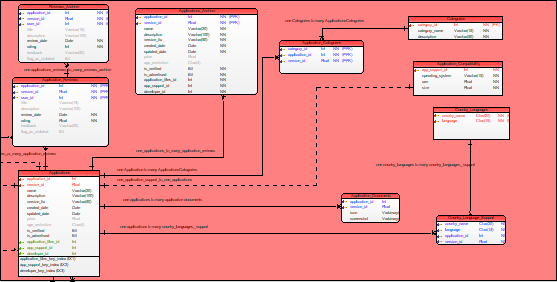


# **Applications Cluster**

There are in all five clusters in this model namely Applications, Downloads, Users, Developers, and Systems. Here we are going to start with the application cluster and its entities. All the entities related with an application like categories, reviews, application support comes under Application cluster.

The following are the entities under Applications Cluster:

*Applications cluster*



## **Applications**

**Definition:** This is the main entity of the Application Cluster. This entity holds the different applications published by the Developers. It holds the versions, created date, verifications of the applications and other attributes.

**Business Rules:**

**In Scope -**

1. The application\_id and version\_id is combine primary key of the table. It can be called as a Composite key.
2. One applications can have many downloads or can have no downloads.
3. An application should have at least one developers associated with it.
4. There should be at least one category per application.
5. There can be zero reviews for an application.
6. There can be no payments for an application when its free.
7. The application must be verified to be screened for users.
8. The applications are archived after 5 versions are entered in the application entity. i.e when there is a 6th version entered in the Application Entity, the 1st version entry will be archived.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **application\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to an application entity when an application is published by the developer for verification. It is a primary key combined with the version\_id known as composite key. The sequence starts with 101. The id is auto incremented.  Example: 101 |
| **version\_id** | Float | Primary Key (PK)  Not Null (NN) | A unique identifier to an application entity when an application is published by the developer. It is a primary key combined with the version\_id known as composite key. The id is not auto incremented.  Example: 1.0 |
| **name** | Varchar (20) | Unique  NN | Name of the application which the developers want to be published.  Example: WhatsApp |
| **description** | Varchar (100) | NN | The description of the application as in what is the use of it.  Example: It is used to connect with people and chat, video call, etc. |
| **version\_fix** | Varchar (50) | NN | It describes the changes in the version from the previous ones or the bug fixes.  Example: contact details bug fixed. |
| **created\_date** | Date | NN | The date of application published by the Developer on the store.  Example: 07/18/2015 |
| **updated\_date** | Date | NN | Date on which the application new version was updated and published.  Example: 05/21/2016 |
| **price** | Float |  | The price of the application in dollars if it is paid otherwise the values is NULL which means it is a free application.  Example: 789.90 |
| **age\_restriction** | Char (4) |  | Age limit to use the application. Above the mention age users can access the application. If NULL then no limits.  Example: 18 |
| **is\_verified** | Bit | NN | It can store 2 values 0 or 1. 0 means not verified and 1 means verified.  Example: 1 |
| **is\_advertised** | Bit | NN | It can store 2 values 0 or 1. 0 means not advertised and 1 means advertised.  Example: 1 |
| **application\_files\_id** | Int | Foreign Key (FK)  NN | It is used to locate the files, images, logos, and screenshots used by the application.  Example: 11 |
| **app\_support\_id** | Int | Foreign Key (FK)  NN | It is used to identify the application compatibility and other information related to support.  Example: 401 |
| **developer\_id** | Int | Foreign Key (FK)  NN | It is used to identify the developer of the application.  Example: 1001 |

**Sample Data:**





## **Applications\_Archive**

**Definition:** It comes under Applications Cluster and stores all the archived applications. The attributes are same as Applications Entity as well as their data type. All the applications will be archived after the sixth version of the application is released to manage the space.

**Business Rules:**

**In Scope -**

1. Only latest 5 versions of the application will be present in the applications tables, all previous other will be archived.

## **Application\_Reviews**

**Definition:** This entity also comes under Application Cluster. This entity holds the reviews of the applications by version and keeps the feedback which can be used determine the application performance in the real world. It is the bridge table between Applications and Users. Also, an user can flag an app inn appropriate using this entity.

**Business Rules:**

**In Scope –**

1. One application review must be linked with both Application and the user.
2. The rating attribute can have maximum of 5.0 as the value and least 1.0.
3. Title, feedback and description attributes can be null as it depends on the users to give a feedback.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **application\_id** | Int | Primary Foreign Key (PFK)  Not Null (NN) | A unique identifier to an application reviews entity along with version\_id and user\_id.  Example: 101 |
| **version\_id** | Float | Primary Foreign Key (PFK)  Not Null (NN) | A unique identifier to an application reviews entity along with application\_id and user\_id.  Example: 1.0 |
| **user\_id** | Float | Primary Foreign Key (PFK)  Not Null (NN) | A unique identifier to an application reviews entity along with application\_id and version\_id.  Example: 10001 |
| **title** | Varchar (15) |  | Title of the review.  Example: Superb!! |
| **description** | Varchar (100) |  | The description of the application reviews given by the users.  Example: It is one of the best application for chatting. |
| **review\_date** | Date | NN | The date of the review was given by the user.  Example: 04/04/2016 |
| **rating** | Float | NN | Ratings given by the user out of 5.0  Example: 3.9 |
| **feedback** | Varchar (50) |  | The feedback given by the user to give some advice to the developers.  Example: The application should include user info in details. |
| **flag\_as\_violated** | Bit |  | It can store 2 values 0 or 1. 0 means not violated and 1 means violated according to the user but there will be a check from the app store as well before flagging the application completely violated (discontinuing the app).  Example: 1 |

**Sample Data:**



## **Reviews\_Archive**

**Definition:** It comes under Applications Cluster and stores all the archived reviews. The attributes are same as Applications Reviews Entity as well as their data type. All the reviews will be archived after the version of the application is not publishing by the app store.

**Business Rules:**

**In Scope –**

1. The reviews will be archive after the application stops publishing that version.

## **Application\_Compatibility**

**Definition:** This entity also comes under Application Cluster. This entity holds the compatibility requirement of the application which is the minimum requirement of a device. If the requirements are not matching the device will not be able to run the application efficiently or it may not install.

**Business Rules:**

**In Scope –**

1. One application can have one application compatibility requirement per app. These details can be updated later when there is a need and the developer is ready to make the changes to the application in the next version.

**Out of scope –**

1. The device\_type specific attributes are out of scope.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **app\_support\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to an application supports entity. The sequence starts with 1231. The id is auto incremented.  Example: 1231 |
| **operating\_system** | Varchar (15) | NN | The operating systems can be ios, windows, android or all.  Example: android |
| **ram** | Float | NN | It minimum ram size required to download the application. It will be in gbs and not mbs.  Example: 2 |
| **size** | Float | NN | It is the size of the application in mbs. If the device is out of memory it cannot download the application.  Example: 45.89 |

**Sample Data:**



## **Categories**

**Definition:** This entity also comes under Application Cluster. This entity holds the categories of the applications. As each application has some features and ultimately leads us to define this entity so that one can distinguish between the types of applications.

**Business Rules:**

**In Scope –**

1. One category can have many applications, however at last one applications must be linked with one category.
2. Categories here are defined based on the features of the applications and the description of the applications.

**Out of scope –**

1. Sub categories are out of scope for this version.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **category\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to a category entity. The sequence starts with 1231. The id is auto incremented.  Example: 201 |
| **category\_name** | Varchar (15) | NN | Name of the category.  Example: gaming |
| **description** | Varchar (50) | NN | The description of the category  Example: it is use to play games of wrestling |

**Sample Data:**



## **Application\_Documents**

**Definition:** This entity holds all the icons and screenshots associated with the applications. It is separate entity as images take a lot of space and hence it is better to keep it in a different entity than applications.

**Business Rules:**

**In Scope –**

1. One application will have atleast one application documents to store the image icon.
2. The image icon may change as per the version and the other screenshots.

**Out of scope –**

1. The videos are not included which can be stored as per the app.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **application\_id** | Int | Primary Foreign Key (PFK)  Not Null (NN) | A unique identifier to an application reviews entity along with version\_id and user\_id.  Example: 101 |
| **version\_id** | Float | Primary Foreign Key (PFK)  Not Null (NN) | A unique identifier to an application reviews entity along with application\_id and user\_id.  Example: 1.0 |
| **icon** | Image | NN | It is the icon of the application which can be changed as per the version cange.  Example: icon.jpg |
| **screenshot** | Image | NN | It is used to store the screenshots which are used in the application.  Example: screenshot1.jpg |

**Sample Data:**



## **Country\_Languages**

**Definition:** This entity stores the data of the country and the languages we are used in communication. One country can have many languages and hence it is and identified entity.

**Business Rules:**

**In Scope –**

1. One application can more than one language in a country.
2. A new version of the application can add new languages and country data.

**Out of scope –**

1. Not all the languages are included in this entity.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **country\_name** | Char (20) | Primary Key (PK)  Not Null (NN) | A composite primary key along with the language to identify many languages in one country.  Example: USA |
| **language** | Char (15) | Primary Key (PK)  Not Null (NN) | A composite primary key along with the country\_name to identify many languages in one country.  Example: spanish |

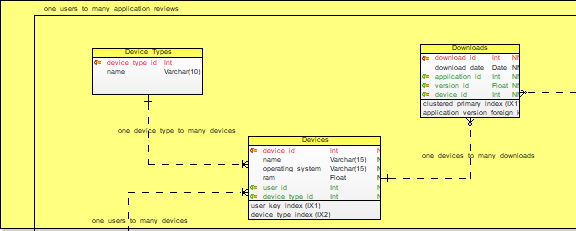
**Sample Data:**



# **Downloads Cluster**

It holds downloads and the devices used by the users to download the applications. An application can be downloaded multiple times by an user and can be downloaded on different devices.

*Downloads Cluster*



## **Downloads**

**Definition:** This entity holds the downloaded applications by the user on a device. It is used to track which application is downloaded on which device and let user know about the updates if available.

**Business Rules:**

**In Scope –**

1. An application can be downloaded multiple times.
2. One device can have many downloads or no downloads, but if there is a download there must be at least one device linked to it.

**Out of scope –**

1. Downloads archive table for the current model is out of scope.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **download\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to a downloads entity. The sequence starts with 11. The id is auto incremented.  Example: 11 |
| **download\_date** | Date | NN | Date when the device downloaded the application.  Example: 09/09/2015 |
| **application\_id** | Int | Foreign Key (FK)  NN | To indicate the application along with version id.  Example: 102 |
| **version\_id** | Float | Foreign Key (FK)  NN | To indicate the application along with application id.  Example: 12.2 |
| **device\_id** | Int | Foreign Key (FK)  NN | Device indication for the downloads  Example: 23 |

**Sample Data:**



## **Devices**

**Definition:** This entity holds the devices used by the users. One can get to know the operating systems and compatibility of the applications based on the devices on which the application is running.

**Business Rules:**

**In Scope –**

1. One device can be used be used by many users.
2. One device can download many applications depending upon the space on the device.
3. One device must have at least one device type associated to it.

**Out of scope –**

1. Device space is out of scope as it can be increased using an external memory card.
2. Devices are not further categorized based on the model number. Name itself hold entire data along with the model number.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **device\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to a device entity. The sequence starts with 21. The id is auto incremented.  Example: 21 |
| **name** | Varchar (15) | NN | The name of the device or the product name.  Example: Samsung A5 |
| **operating\_system** | Varchar (15) | NN | The operating systems can be ios, windows, android or all.  Example: android |
| **version** | Float | NN | To indicate the application along with version id.  Example: marshmallow |
| **user\_id** | Int | Foreign Key (FK)  NN | To indicate the user of the device.  Example: 10001 |
| **device\_type\_id** | Int | Foreign Key (FK)  NN | To identify the device type of the device.  Example: 31 |

**Sample Data:**



## **Device\_Types**

**Definition:** This entity holds the device type used by the users. One can get to know whether the device is mobile or an ipad.

**Business Rules:**

**In Scope –**

1. One device type can have many devices.

**Out of scope –**

1. Desktop devices are currently not in scope.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **device\_type\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to a device type entity. The sequence starts with 31. The id is auto incremented.  Example: 31 |
| **name** | Varchar (10) | NN | The name of the device\_type or the product\_type name.  Example: Ipad |

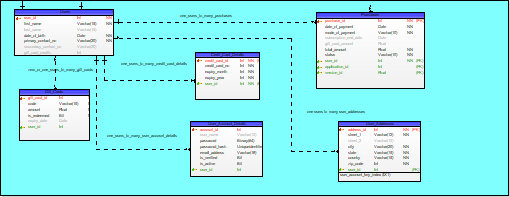
**Sample Data:**



# **Users Cluster**

It holds all the user details including his account details, his credit card details, personal information, purchases of the applications, etc. It is made available to track the application process through customers.

*Users Cluster*



## **Users**

**Definition:** This entity comes under Users Cluster. It is used to keep the user’s basic information which he can use to sign up on any application. One can get to know the age of the user through this entity and can verify the user for the age restriction which are different with different applications.

**Business Rules:**

**In Scope –**

1. A user can have many devices; hence a user can have many downloads. However, a user as to be linked at least one device.
2. A user can have zero or many accounts.
3. A user can have zero or many addresses.
4. A user can have zero or many gift cards but cannot use the same gift card again.
5. A user can have zero or many credit cards.
6. A user can have zero or many purchases of the applications.
7. A user can give zero or many application reviews, however only once each version of the applications.
8. The last name and the secondary contact no attributes can be null as it is optional.

**Out of scope –**

1. The Users localization is out scope.
2. Friends suggestion is out of scope.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **user\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to a User Entity. The sequence starts with 10001. The id is auto incremented.  Example: 10001 |
| **first\_name** | Varchar (15) | Not Null (NN) | The first name of the user using the device.  Example: Peter |
| **last\_name** | Varchar (15) |  | The last name of the user using the device.  Example: Shaw |
| **date\_of\_birth** | Date | Not Null (NN) | The date of birth is not null since we need to keep a track on the age of the user.  Example: 03/04/1994 |
| **primary\_contact\_no** | Varchar (20) | Not Null (NN) | The contact number, could be a mobile number.  Example: 8168375673 |
| **secondary\_contact\_no** | Varchar (20) |  | The contact number, could be a mobile number.  Example: 9168775673 |
| **gift\_card\_credis** | Int |  | It is the amount (say in dollars) which was redeemed through a gift card and can be used further for any application purchase.  Example: 20 |

**Sample Data:**



## **User\_Account\_Details**

**Definition:** Since every user has an account if the application requires a user to create an account. Hence to keep a track on the accounts of the user we have created this entity in this model.

**Business Rules:**

**In Scope –**

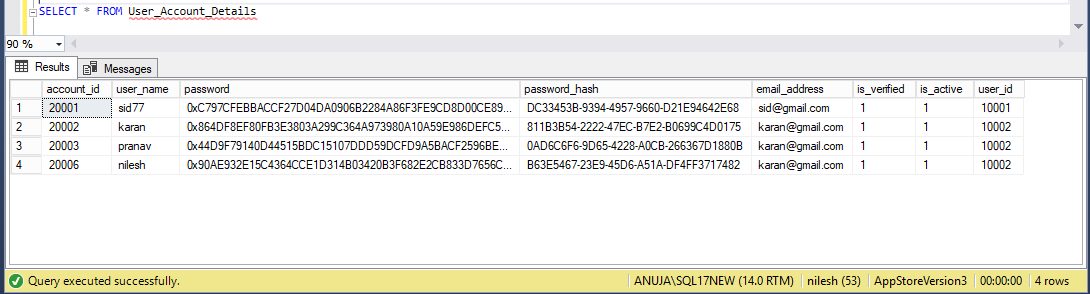
1. One user can have many accounts.
2. If the user name is null email address is considered as the user name.
3. The account is first verified by the app store since the account might have already created hence only if is verified attribute is 1 account is able to access the applications and use.
4. The accounts status can be tracked through is active attribute. If the value is deactivated the user can still activate the account. But if the account is deleted the user cannot sign in with same credentials

**Out of scope –**

1. The deleted accounts can be archived.
2. Password encryption.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **account\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to an account details entity. The sequence starts with 20001. The id is auto incremented.  Example: 20001 |
| **user\_name** | Varchar (12) |  | The user name used to login into an application  Example: nilkhs77 |
| **password** | Binary (64) | NN | The password used to login into an application. It is encrypted.  Example: 123 |
| **Password\_hash** | UNIQUEIDENTIFIER() | NN | A new unique number for encryption of password.  Example : NEWID() |
| **email\_address** | Varchar (18) | NN | Email address of the user  Example: nikhil77@gmail.com |
| **is\_verified** | Bit | NN | Used to check whether the account is verified or not. Can have two values 1 if verified.  Example: 1 |
| **is\_active** | Bit | NN | Used to check whether the account is active or not. Can have two values 1 if active.  Example: 1 |
| **user\_id** | Int | Foreign Key (FK)  NN | It is used to identify the users.  Example: 10028 |

**Sample Data:**



## **Purchases**

**Definition:**  It is used keep a track of the purchased application by the users and the payments information. It will keep a track of the application payment status and the user can get the information of their transactions.

**Business Rules:**

**In Scope –**

1. One user can have zero or many payments.
2. If there is a payment it must be linked with at least one application.
3. If there is payment, then it can be entirely paid through a gift or via gift card and online payments which further might be linked to a credit card.
4. The gift card amount can be null.
5. Mode of payment be only gift card, online payment or both. Both indicates gift card plus online payment using a credit card.
6. If the amount is zero, then the application is a free app.
7. If the subscription end date is null, then the subscription is life time if amount is not zero.

**Out of scope –**

1. The user can also get the money back in case the application has that feature to return the money back.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **purchase\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to any transactions between purchases and a user. The sequence starts with 60021. The id is auto incremented.  Example: 60021 |
| **date\_of\_payment** | Date | NN | The date on which the payment was initiated.  Example: 1/1/2015 |
| **mode\_of\_payment** | Varchar (10) | NN | It can be gift card payment or online payment using a credit card or both.  Example: both |
| **subscription\_end\_date** | Date |  | The end date can be null if the app is free or the subscription is lifetime.  Example: 1/1/2015 |
| **gift\_card\_amount** | Float |  | The amount used in the transaction of the gift card credits attribute of the user.  Example: 20 |
| **total\_amount** | Float | NN | The total amount of the transaction  Example: 400 |
| **status** | Varchar (10) | NN | The status of the transaction whether it is initiated or successful or in the intermediate state.  Example: initiated |
| **user\_id** | Int | Foreign Key (FK)  NN | Users identifier.  Example: 1001 |
| **application\_id** | Int | Foreign Key (FK)  NN | Application identifier.  Example: 32202 |
| **version\_id** | Float | Foreign Key (FK)  NN | Applications identifier.  Example: 32.0 |

**Sample Data:**



## **Credit\_Card\_Details**

**Definition:** This entity used to store the credit card details which can be used by a user if he or she purchases any application.

**Business Rules:**

**In Scope –**

1. One credit card can only be linked to one user.
2. User can have zero or many credit cards.

**Out of scope –**

1. Debit cards used by the user.
2. Bank details of the user.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **credit\_card\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to Credit card details entity. The sequence starts with 109. The id is auto incremented.  Example: 101 |
| **credit\_card\_no** | Int | NN | The credit card number of the user.  Example: 9836472537 |
| **expiry\_month** | Int | NN | The expiry month of the card.  Example: 12 |
| **expiry\_year** | Int | NN | The expiry year of the card.  Example: 2022 |
| **user\_id** | Int | Foreign Key (FK)  NN | Users identifier.  Example: 1001 |

**Sample Data:**



## **Gift\_Cards**

**Definition:** This entity stores different gift cards with their respective codes and also keep a track on which card has been redeemed and which is not.

**Business Rules:**

**In Scope –**

1. One gift can be linked to only one user.
2. It cannot be used once it is redeemed.
3. The redeemed amount is added to the gift card credits attribute of the users entity.
4. There can be no expiry to a gift card. In this case, the expiry date is null.

**Out of scope –**

1. Gift cards which are redeemed should be archive or not.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **gift\_card\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to a gift card. The sequence starts with 103. The id is auto incremented.  Example: 103 |
| **code** | Varchar (15) | NN | A code which user can enter to redeem the amount to his account.  Example: 1SDSA84SDREW |
| **amount** | Float | NN | A amount of the gift card, a gift card value.  Example: 500 |
| **is\_redeemed** | Bit | NN | Whether the gift card is redeemed or not. 1 means yes redeemed, 0 means not redeemed.  Example: 0 |
| **expiry\_date** | Date |  | The gift card can be expired by a date. This date can be expiry date  Example: 01/02/2027 |
| **user\_id** | Int | Foreign Key (FK)  NN | Users identifier.  Example: 1001 |

**Sample Data:**



## **User Addresses**

**Definition:** This entity is used to store the address of the users as well as developers. However, the users should ideally have an address and the developers may not have an address. It is debatable; hence it comes under Users cluster in this version.

**Business Rules:**

**In Scope –**

1. One user can have zero or many addresses.
2. Country is mandatory field as it can be used to get the compatibility of the application and we can check availability with respect to the application.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **address\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to the Addresses Entity. The sequence starts with 101. The id is auto incremented.  Example: 101 |
| **street\_1** | Varchar (15) | NN | Street name of the address  Example: Boylston St |
| **street\_2** | Varchar (15) |  | Street name of the address  Example: apt 238 |
| **city** | Varchar (15) | NN | The city where user lives  Example: Boston |
| **state** | Varchar (100) | NN | The state of the user address.  Example: MA |
| **country** | Varchar (15) | NN | Country of the address  Example: USA |
| **zip\_code** | Int | NN | Zip code of the address  Example: 22993 |
| **user\_id** | Int | Foreign Key (FK)  NN | Users identifier.  Example: 1001 |

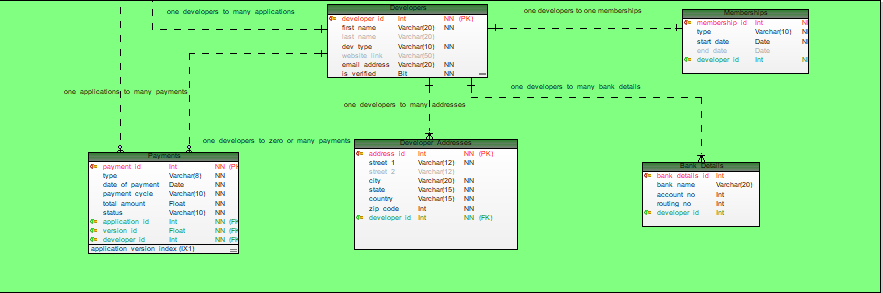
**Sample Data:**



# **Developers Cluster**

It holds all the developer details including his bank details, personal information, payments of the applications, memberships, etc. It is made available to track the application process through developers.

*Developers Cluster*



## **Developers**

**Definition:** This entity is used to store the developers details of the applications. It also stores the type of developers and check if the user is verified or not.

**Business Rules:**

**In Scope –**

1. One developer can have many applications.
2. One developer can have only one membership at a time.
3. One developers can have one or many bank details.
4. One developers can have zero or many addresses.
5. One developer can have zero or no payments.
6. Only verified user will be able to publish the applications on the app store.

**Out of scope –**

1. Enterprise developers are out of scope.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **developer\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to the Developers Entity. The sequence starts with 1001. The id is auto incremented.  Example: 1001 |
| **first\_name** | Varchar (20) | Not Null (NN) | The first name of the developer.  Example: Peter |
| **last\_name** | Varchar (20) |  | The last name of the developer.  Example: Shaw |
| **dev\_type** | Varchar (10) | NN | The type of developer either an individual or an organization.  Example: organization |
| **website\_link** | Varchar (50) |  | The website of the organization or the individual if he or she has.  Example: www.googlegames.com |
| **email\_address** | Varchar (20) | NN | The email address of the individual or company.  Example: rghjitr@regamer.com |
| **is\_verified** | Bit | NN | If values is 1 developer is verified otherwise the value is 0 which not verified.  Example: 1 |

**Sample Data:**



## **Payments**

**Definition:**  It is used keep a track of the payments of an application by the app store to the developers and the payments information. It will keep a track of the payment status and the developers can get the information of their transactions.

**Business Rules:**

**In Scope –**

1. One developer can have zero or many payments.
2. If there is a payment it must be linked with at least one application.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **payment\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to any transactions between payments and a developer. The sequence starts with 10029. The id is auto incremented.  Example: 10029 |
| **type** | Varchar (8) | NN | It is used to define whether the payment is from developer or to developer.  Example: to dev |
| **date\_of\_payment** | Date | NN | The date on which the payment was initiated.  Example: 1/1/2015 |
| **payment\_cycle** | Varchar (10) | NN | It can be monthly quarterly or yearly.  Example: monthly |
| **status** | Varchar (10) | NN | The status of the transaction whether it is initiated or successful or in the intermediate state.  Example: initiated |
| **total\_amount** | Float | NN | The total amount of the transaction  Example: 400 |
| **developer\_id** | Int | Foreign Key (FK)  NN | Developers identifier.  Example: 10035 |
| **application\_id** | Int | Foreign Key (FK)  NN | Application identifier.  Example: 3220 |
| **version\_id** | Float | Foreign Key (FK)  NN | Applications identifier.  Example: 32.0 |

**Sample Data:**



## **Memberships**

**Definition:** This entity used to store different types of memberships and the developers linked to it. Developer should have a membership to publish an application on the app store.

**Business Rules:**

**In Scope –**

1. One developer can have only one membership at a time.
2. Memberships end date can be null which means the membership is lifetime.

**Out of scope –**

1. Enterprise memberships out of scope.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **membership\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to Memberships entity. The sequence starts with 1022. The id is auto incremented.  Example: 1022 |
| **type** | Varchar (10) | NN | The type of membership: premium, business, golden, etc  Example: premium |
| **start\_date** | Int | NN | The start date of the membership.  Example: 12/2/2017 |
| **end\_date** | Int |  | The expiry year of the membership.  Example: 2/2/2022 |
| **developer\_id** | Int | Foreign Key (FK)  NN | ddeveloper identifier.  Example: 10021 |

**Sample Data:**



## **Bank\_Details**

**Definition:** This entity used to store the bank details which can be used by payments entity if he or she get payments from the app store.

**Business Rules:**

**In Scope –**

1. One bank details can be linked to only one developer.
2. Developers can have many bank accounts.

**Out of scope –**

1. Swiss code for international banking.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **bank\_details\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to Bank details entity. The sequence starts with 10002. The id is auto incremented.  Example: 10002 |
| **bank\_name** | Varchar (20) | NN | The name of the bank.  Example: BOA |
| **account\_no** | Int | NN | The bank account number.  Example: 76583242343 |
| **routing\_no** | Int | NN | The bank routing number.  Example: 324223442 |
| **developer\_id** | Int | Foreign Key (FK)  NN | Developers identifier.  Example: 17061 |

**Sample Data:**



## **Developer Addresses**

**Definition:** This entity is used to store the address of the developers. The developer may or may not require storing addresses. But if there is a requirement the developer addresses are stored in this entity.

**Business Rules:**

**In Scope –**

1. One developer can have zero or many addresses.
2. Country is mandatory field as it can be used to get the application developed with respect to countries.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **address\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to the Addresses Entity. The sequence starts with 101. The id is auto incremented.  Example: 101 |
| **street\_1** | Varchar (15) | NN | Street name of the address  Example: Boylston St |
| **street\_2** | Varchar (15) |  | Street name of the address  Example: apt 238 |
| **city** | Varchar (15) | NN | The city where user lives  Example: Boston |
| **state** | Varchar (100) | NN | The state of the user address.  Example: MA |
| **country** | Varchar (15) | NN | Country of the address  Example: USA |
| **zip\_code** | Int | NN | Zip code of the address  Example: 22993 |
| **user\_id** | Int | Foreign Key (FK)  NN | Users identifier.  Example: 1001 |
| **developer\_id** | Int | Foreign Key (FK)  NN | Developer identifier.  Example: 1056 |

Sample Data:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **address\_id** | **street\_1** | **street\_2** | **city** | **state** | **country** | **zip\_code** | **developer\_id** |
| 101 | Boylston St | Apt 238 | Boston | MA | USA | 2283 | 1001 |
| 102 | germain St | Apt 239 | New York | MA | USA | 2001 | 1002 |
| 103 | Boylston St | Apt 240 | Boston | MA | USA | 2283 | 1003 |
| 104 | germain St | Apt 241 | Boston | MA | USA | 2001 | 1004 |
| 105 | germain St | Apt 242 | Boston | MA | USA | 2984 | 1005 |

# **Systems Cluster**

It stores the data of applications and other metadata.

*Systems Cluster*



## **Application\_Files**

**Definition:** This entity used to store the meta data of the applications entity. It stores the path to the various files used by the application.

**Business Rules:**

**In Scope –**

1. One application must have at least one application files.
2. The path in the entity stores everything including .akp or .ios file, screenshots, images, etc. No separate path for separate files

**Out of scope –**

1. Employees of the app store

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **application\_files\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to Application Files entity. The sequence starts with 10031. The id is auto incremented.  Example: 10031 |
| **path** | Varchar (50) | NN | The path to the files folder of the local disk for now.  Example: C:\Users\Desktop\IS\DBDS |

**Sample Data:**



## **Admin Credentials**

**Definition:** This entity used to store admin credentials.

**Business Rules:**

**In Scope –**

1. At least one admin should be present for managing the applications.

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute Name** | **Data Type** | **Constraints** | **Definition and example** |
| **credentials\_id** | Int | Primary Key (PK)  Not Null (NN) | A unique identifier to the table. The sequence starts with 301.  Example: 1091 |
| **username** | Varchar (10) | NN | The basic username which the admin will get  Example: admin |
| **password** | Varchar (15) | NN | Stores password of the admin which can be of 15 alphanumeric value  Example: admin123 |

**Sample Data:**



# **Relationship Description**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table 1** | **Table 2** | **Relationship** | **Identified/ Non-Identified** | **Description** |
| Applications | Application\_Documents | one to many | Non\_Identified | One application will have atleast one or many icons which needs to be store. The application screenshots or icon may change as per the version change |
| Applications | Country\_Languages | many to many | Identified | One application can support more than one languages and can be published in different countries. Similarly, one country can have many applications published in their country |
| Applications | Categories | many to many | Identified | One applications will have atleast one or many categories. Similarly, one category will have atleast one or many applications related to it. |
| Applications | Application\_Compatibility | one to one | Identified | One application will have one application compatibility requirement, which be a minimum requirement. |
| Applications | Application\_Reviews | one to many | Identified | One applications will have zero or many application reviews. Since, it must be unique for each version relationship is identified. |
| Applications | Applications\_Archive | one to many | Identified | Since application archive cannot be independent, it is identified. One application will have zero or many archive entries. |
| Applications | Payments | one to many | Non\_Identified | One application will have zero or many payments. Zero if the app is not downloaded or is not yet verified. |
| Applications | Purchases | one to many | Non\_Identified | One application will have zero or many payments. Zero if the app is not yet verified. |
| Applications | Downloads | one to many | Non\_Identified | One application will have zero or many downloads. Zero if the app is not downloaded or is not yet verified. But downloads will be linked to atleast one application. |
| Applications | Application\_Files | one to one | Non\_Identified | One application will have one application files related to it which stores the application metadata. |
| Developers | Applications | one to many | Non\_Identified | One developer will have atleast one or many applications. A developer can create more than one application. |
| Developers | Memberships | one to one | Non\_Identified | One developer will have one membership linked to it so that he can be verified. |
| Developers | Bank\_Details | one to many | Non\_Identified | One developer will have atleast one or many bank accounts. Atleast one is needed to get the purchase amount transferred from the app store. |
| Developers | Payments | one to many | Non\_Identified | One developer will have zero or many payments. Zero if the application is still not verified. |
| Developers | Developer\_Addresses | one to many | Non\_Identified | One developer will have atleast one or many addresses. A developer can create more than one address. |
| Devices | Downloads | one to many | Non\_Identified | One device will have zero or many downloads. Zero if the app is not yet downloaded. |
| Device\_Types | Devices | one to many | Non\_Identified | One device type will have atleast one or many devices. There can be multiple types of devices. |
| Users | Devices | one to many | Non\_Identified | One user will have atleast one or many devices. Atleast one because user cannot be defined without a device. |
| Users | Application\_Reviews | one to many | Non\_Identified | One user will have zero or many application reviews. Zero if the user is not interested to review the application and many if there is a new version for the same application. |
| Users | Gift\_Cards | one to many | Non\_Identified | One user will have zero or many gift cards. Zero if the user is linked to any gift card. |
| Users | User\_Account\_Details | one to many | Non\_Identified | One user will have zero or many accounts. Zero if the application does not require any account. |
| Users | Credit\_Card\_Details | one to many | Non\_Identified | One user will have zero or many credit cards. Zero if the application does not require any address details. |
| Users | User\_Addresses | one to many | Non\_Identified | One user will have zero or many addresses. Zero if the user is linked to any gift card. |
| Users | Purchases | one to many | Non\_Identified | One user will have zero or many purchases. Zero if the user has not downloaded any application but searched. |

# **Related Documents**

# **Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version** | **Author** | **Last Revised Date** | **Comments** | **Reviewer** | **Future Scope** |
| 1.0 | Nilesh Nerkar | 2/2/2018 | Major Entities created along with their basic definitions. | Priyal Chaudhari | More entities can be added |
| 1.1 | Nilesh Nerkar | 2/13/2018 | Draft 2 created. Major entities added. Relationships defined in the documents. | Akshay Jain | Relationship between the entities can be created using a data model tool and hence the DDL script for the database. Entities can be added. Sample data can be added. |
| 1.2 | Nilesh Nerkar | 3/2/2018 | App store Data model created using TOAD Data Modeler. Major entities like purchases, devices, downloads, memberships, addresses, payment details, account details, application metadata, reviews application support, etc added. Sample data has been added. | Akshay Jain | Sample data can be inserted into the database. Relationships between the entities can be improved. Indexes and stored procs can be included. |
| 2.0 | Nilesh Nerkar | 4/6/2018 | App Store Version 2 created which included 25 entity classes, relationships which were incorrect are modified. In App Advertisement, geography, languages, subscription, application compatibility, reviews archive are added. Also, indexes are included in the current model. | Akshay Jain | Stored Procedures and triggers can be included for the performance. A basic user interface can be created to test the database functionality. |
| 3.0 | Nilesh Nerkar | 4/22/2018 | Updated the password data type to binary and added a password\_has column with uniqueidentifier datatype. Added Admin Credentials table to store admin username and password. Update the icon and screenshot datatype to VARBINARY.  Included three stored procedures to insert applications which validates unique application name, one to encrypt the password and the last to decrypt the password field.  Included views to get the rating of application and added trigger to archive the application after the fifth count.  Applied indexes on required fields, included sequences on each primary key to auto increment the value. | Akshay Jain | Application database can be backed up using a stored procedure.  An identity must set on primary key  Invoice can be generated |