

# Database design of the application store

**Submitted by**

Sami Ranjan Shekhar

**Under the guidance of**

Prof. Vincent Lattuada

# Database Design of an Application Store

The Application Store is an online portal that is provided by the phone OS company(Android store/Apple store) through which applications can be downloaded by the users on their devices (mobile phones, tablets) based on different factors such as support, compatibility etc.

The application store can be divided into several major clusters. The clusters considered for this app store are applications, downloads, users, developer and payment.

## Application Store Major Entities

### *Cluster: Applications*

The application cluster would consist of 4 tables: Applications, Category, Version, Ratings and reviews.

### Application

- ✓ The Application table consists all the primary details of the application. It holds the names of the applications, application id along with a short description.
- ✓ The category of the application and the version of the app. It could have some screenshots of the user interface of the application, to have an overview of what the app would look like. There would be a column with the developer name/Company name.
- ✓ One to many relationships between application and the developer can be made.

Business Rules:

- An application has to have a developer associated with it, though it can also have multiple developers associated.
- The application has to be within one category.

Name	Data Type	Define	Example	Null/key
AppID	VarChar(10)	A numeric ID assigned to the application	45444	NN/PK

DeveloperID	VarChar(10)	A numeric ID assigned to the developer	54567	NN/FK
AppName	VarChar(20)	Name of the application	Ashphalt 8	NN
AvgUserRating	Float	Avg Rating of the application(It should be between 1 and 5)	4.4	NN
CategoryID	VarChar(10)	Category of the application	111	NN/FK
DateCreated	Date	Date the app was created	05/08/2016	NN
DateLastUpdated	Date	Date the app was last updated	08/08/2016	NN
AppDesc	Varchar(250)	A short app description	A thrilling shooting game.	NN
Compatiility	Varchar	This defines the OS version and above that supports the app	IOS 10 and above	NN
AppSize	VarChar	Size of the app	65 MB	NN
AgeRating	SmallInt	Age group the app is intened to be used by	17	NN
DownCount	INT	No of downloads	54433	NN

### Category

- ✓ This table will contain the type/genre of the application and will help to fetch all the applications in the same category with ease.
- ✓ The Category and Application will have a one to many relationship as one category can have multiple applications.

### Business Rules:

- Every application must have at least one category associated with it.

- Every Category must have at least one application associated with it.
- Only the author of an application can define the category it is associated with.

Name	Data Type	Define	Example	Null/Key
CategoryID	VarChar(10)	A unique category ID assigned to every category	111	NN/PK
CategoryName	VarChar(20)	Category of the application	Gaming	NN
CategoryDesc	VarChar(50)	NN/FK	Immerse in the gaming	

### Version

- ✓ The version of the downloaded application
- ✓ Can be connected to the users and the application table to show the version of the app that the user has downloaded

### Business Rules

- A version of the app should be unique when connected to the user
- Version can only be released by the Author of the application.
- Two different versions of an application can exist on two different devices associated with a single user.

Name	Data Type	Define	Example	Null/Key
VersionID	Varchar(10)	A numeric ID given to the version	4243	NN/PK
VersionName	VarChar(10)	The version Number	1.1	NN
ReleaseDate	Date	The date of the release of the version	09/08/2015	NN

AppID	VarChar(10)	Application ID of the application the version is associated with	45445	NN/FK
-------	-------------	--	-------	-------

### Ratings and Reviews

- ✓ It can also contain a review column for the app reviews provided by the users.
- ✓ Also it will be linked to the application table through the application ID.

#### Business Rules:

- One user can provide only one rating for an application.
- A rating ID which will contain the rating and it will be linked to the user table through UserID and Application table through AppID
- This table can contain a rating column for each application where a constraint can be added so as to accept only integer values from 1 to 5.
- Only the users who have downloaded the application can give a rating

Name	Data Type	Define	Example	Null/Key
Rating ID	VarChar(10)	Assigned to rating with a unique number	46	NN/PK
UserID	VarChar(10)	User detail about the user who provided the rating	67777	NN/FK
AppID	VarChar(10)	Application for which the rating was given by the user	45444	NN/FK
Reviews	Varchar(300)	Review by the user	Great game	
DateCreated	Date	Date on which the rating was given by the user	08/06/2017	NN

UserRating	Float	Rating given by the user(1 to 5)	4	
------------	-------	----------------------------------	---	--

## Cluster: Developer

### Developer

- ✓ It will contain the application's developer details. It can contain fields like Developer name, developer ID and similar applications that have been designed by the same developer.
- ✓ As stated earlier it would be linked to the application table.

#### Business Rules:

- A developer can have multiple apps to his name but it can't have 0 apps to this name.

Name	Data Type	Define	Example	Null/Key
DeveloperID	VarChar(10)	An ID that is assigned to the developer.	756	NN/PK
DeveloperName	VarChar(20)	Name of the developer on the app store	Gameloft	NN
AppCount	INT	Number of apps developed by the developer	34	
DevBankID	VarChar(10)	ID to Identify the Bank details of the developer	675767	NN/FK

### Developer Bank Details

- ✓ Stores the bank details of the developer and has restricted access.

Name	Data Type	Define	Example	Null/Key
DevBankID	VarChar(10)	ID to Identify the Bank details of the developer	675767	NN/PK

DevAddress	Varchar(100)	Address of the Developer	114 Boylston street, Boston	NN
DevRoutingNo	INT	Bank routing No of the developer	45365555	NN
DeveloperAccNo	INT	Acc No of the developer	876865453	NN

### Cluster: Users

#### Users

- ✓ User table will contain user details and also it will have the data for all the apps that the user has downloaded.
- ✓ It will have columns like Username, User ID as well as all the invoices that are associated with the user and the apps purchased/downloaded.
- ✓ User ID can be associated with the invoice ID in the invoices tale.

#### Business Rules:

- The password has to be protected by the app store regulators and can't be shared
- Users must have an account with a UserID to download an application.
- User must provide email and payment details to create an account.

Name	Data Type	Define	Example	Null/Key
UserID	VarChar(10)	A unique identifier for the user	101	NN/PK
Fname	VarChar(20)	The first name of the user	Sami	NN
Lname	VarChar(20)	The last name of the user	Ranjan	NN
EmailID	VarChar(20)	Email ID of the user	Ranjansami@yahoo.com	NN
Contact	Char	Contact Number of the user	8882229274	NN

Address	VarChar(50)	This will contain the address of the user	60 saint germain Street, Boston MA- 02115	NN
Password	VarChar(20)	This is contain the password that will help the user to login and will only be accessed by the appstore	Possible123	NN

### User Bank Details

- ✓ Consists of the bank details of the users, including Account Number, Routing number, card no and expiry date.

Name	Data Type	Define	Example	Null/Key
UserBankName	VarChar(20)	The name of the user banks name	Bank of America	NN
UserAccountNo	INT	The users account number	5546336676	NN
UserBankRoutingno	INT	The routing number of the bank	244421	NN
UserCardNo	INT	The card number of the user	4335543356675543	NN
CardExpDate	Date	Card Expiry Date	01/02	NN

### Device

- ✓ The device associated with the app store
- ✓ It can be joined with the user ID so as to keep a track of the users downloads

### Business Rules:

- The device ID should be unique



- It will specify the apps on that unique device
- Any Application can be downloaded to a device only if its compatibility matches with the device.

Name	Data Type	Define	Example	Null/Key
DeviceID	VarChar(10)	A numeric ID given to the Device	3533	NN/PK
UserID	VarChar(10)	The user ID associated with the device	4353	NN/FK
DeviceName	VarChar(20)	Name of the device	Sami's Iphone	NN
Devicetype	VarChar(20)	Model of the device	Iphone x	NN

### Cluster: Payment

#### Invoices

- ✓ This table will contain the purchase details associated to each user ID. The purpose of the table is to hold the transaction details of the purchases made on the app store.
- ✓ This table can include columns like Invoice ID, App ID (linked to the application table), Date of payment etc.

#### Business Rules:

- This will connect the userID who have purchased the apps with the appID.
- Mode of payment can only be credit card or debit card.

Name	Data Type	Define	Example	Null/Key
InvoiceID	VarChar(10)	Transaction ID is a unique identifier for the transaction	576487	NN/PK

UserID	VarChar(10)	This will link the user/Developer tables to application table	101	NN/FK
PaymentType	VarChar(10)	Made by user or the developer. Further the details can be taken out from the User or the developer table.	user	NN
PaymentDate	Date	Date of payment when the app was purchased	09/08/2017	NN
PaymentAmount	Money	The payment amount for the application	\$4	NN
PaymentMode	VarChar(1)	Will contain 2 values either C or D for a credit or a debit card	C	NN

### *Cluster: Downloads*

#### Downloads

- ✓ This table will contain the number of downloads according to the application ID.
- ✓ It will also specify the version downloaded.
- ✓ This will also specify the device to which the app has been downloaded through the device ID and the user through user ID

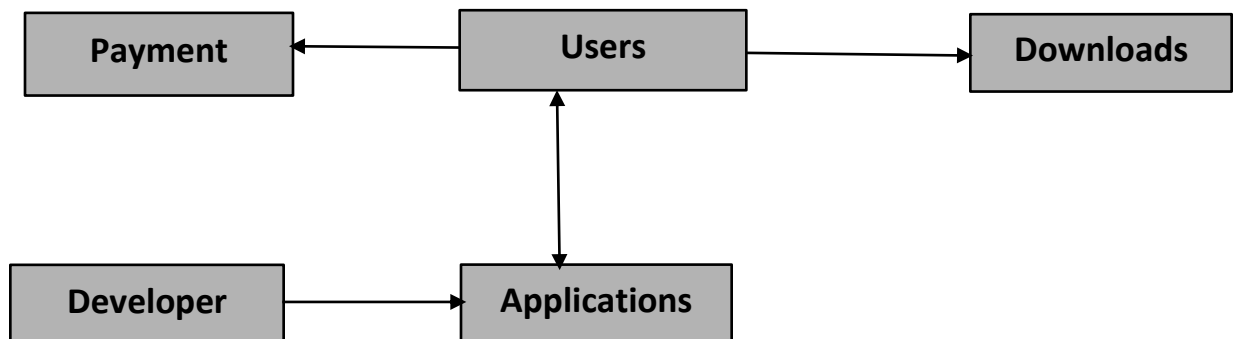
#### **Business Rules:**

- The downloads table will be linked to the user and the devices table to specify the device on which it was downloaded and the user.
- There can be different entries for the same app when different versions are downloaded to find out which users updated it and who did not.

- Any device or user can download an application multiple number of times.
- An updated version can be downloaded once its available.

Name	Data Type	Define	Example	Null/Key
DownloadID	VarChar(10)	Unique ID defining the download instnace	5658	NN/PK
AppID	VarChar(10)	App that was downloaded	Ashphat 8	NN/FK
UserID	VarChar(10)	User who downloaded the application	101	NN/FK
DownloadDate	Date	Date of the download	05/08/2016	NN
VersionName	VarChar(10)	Version of the application downloaded	1.1	NN

### High level cluster diagram



## The DDL script Generated by the TOAD Data Modeler:

```
/*
Created: 4/5/2018
Modified: 4/6/2018
Project: Application Store
Model: Microsoft SQL Server 2016
Author: Sami Ranjan Shekhar
Version: 2
Database: MS SQL Server 2016
*/

-- Create tables section -----

-- Table Application

CREATE TABLE [Application]
(
  [AppID] Varchar(10) NOT NULL,
  [DeveloperID] Varchar(10) NOT NULL,
  [AppName] Varchar(20) NOT NULL,
  [CategoryID] Varchar(10) NOT NULL,
  [AvgUserRating] Float NOT NULL,
  [Downcount] Int NULL,
  [AppDesc] Varchar(255) NOT NULL,
  [DateCreated] Date NOT NULL,
  [DateLastUpdated] Date NOT NULL,
  [AppSize] Varchar(10) NOT NULL,
  [AgeRating] Smallint NOT NULL,
  [Compatibility] Varchar(50) NOT NULL
)
go

-- Add keys for table Application

ALTER TABLE [Application] ADD CONSTRAINT [Key1] PRIMARY KEY ([AppID])
go

ALTER TABLE [Application] ADD CONSTRAINT [AppName] UNIQUE CLUSTERED ([AppName])
go

ALTER TABLE [Application] ADD CONSTRAINT [AppID] UNIQUE CLUSTERED ([AppID])
go

ALTER TABLE [Application] ADD CONSTRAINT [CategoryID] UNIQUE CLUSTERED ([CategoryID])
go

ALTER TABLE [Application] ADD CONSTRAINT [DeveloperID] UNIQUE CLUSTERED ([DeveloperID])
go

-- Table Category

CREATE TABLE [Category]
(
  [CategoryID] Varchar(10) NOT NULL,
```

```

[CategoryName] Varchar(20) NOT NULL,
[CategoryDesc] Varchar(50) NOT NULL
)
go

-- Add keys for table Category

ALTER TABLE [Category] ADD CONSTRAINT [Key2] PRIMARY KEY ([CategoryID])
go

ALTER TABLE [Category] ADD CONSTRAINT [CategoryID] UNIQUE CLUSTERED ([CategoryID])
go

ALTER TABLE [Category] ADD CONSTRAINT [CategoryName] UNIQUE CLUSTERED ([CategoryName])
go

-- Table Developer

CREATE TABLE [Developer]
(
    [DeveloperID] Varchar(10) NOT NULL,
    [DeveloperName] Varchar(20) NOT NULL,
    [AppCount] Int NULL,
    [DevBankID] Varchar(10) NOT NULL
)
go

-- Create indexes for table Developer

CREATE INDEX [IX_Relationship17] ON [Developer] ([DevBankID])
go

-- Add keys for table Developer

ALTER TABLE [Developer] ADD CONSTRAINT [Key3] PRIMARY KEY ([DeveloperID])
go

ALTER TABLE [Developer] ADD CONSTRAINT [DeveloperID] UNIQUE CLUSTERED ([DeveloperID])
go

ALTER TABLE [Developer] ADD CONSTRAINT [DeveloperName] UNIQUE CLUSTERED ([DeveloperName])
go

ALTER TABLE [Developer] ADD CONSTRAINT [DevBankID] UNIQUE CLUSTERED ([DevBankID])
go

-- Table Ratings

CREATE TABLE [Ratings]
(
    [RatingID] Varchar(10) NOT NULL,
    [AppID] Varchar(10) NOT NULL,
    [DateCreated] Date NOT NULL,
    [UserRating] Int NULL,
    [Reviews] Varchar(50) NULL,
    [UserID] Varchar(10) NOT NULL
)
go

```

```

-- Create indexes for table Ratings

CREATE INDEX [IX_Relationship11] ON [Ratings] ([UserID])
go

CREATE INDEX [IX_Relationship12] ON [Ratings] ([AppID])
go

-- Add keys for table Ratings

ALTER TABLE [Ratings] ADD CONSTRAINT [Key4] PRIMARY KEY ([RatingID])
go

ALTER TABLE [Ratings] ADD CONSTRAINT [AppID] UNIQUE CLUSTERED ([AppID])
go

ALTER TABLE [Ratings] ADD CONSTRAINT [UserID] UNIQUE CLUSTERED ([UserID])
go

-- Table Users

CREATE TABLE [Users]
(
    [UserID] Varchar(10) NOT NULL,
    [UserBankID] Varchar(10) NOT NULL,
    [Fname] Varchar(20) NOT NULL,
    [Lname] Varchar(20) NOT NULL,
    [EmailID] Varchar(20) NOT NULL,
    [Contact] Char(10) NOT NULL,
    [Address] Varchar(50) NOT NULL,
    [Password] Varchar(20) NOT NULL
)
go

-- Create indexes for table Users

CREATE INDEX [IX_Relationship18] ON [Users] ([UserBankID])
go

-- Add keys for table Users

ALTER TABLE [Users] ADD CONSTRAINT [Key5] PRIMARY KEY ([UserID])
go

ALTER TABLE [Users] ADD CONSTRAINT [Contact] UNIQUE CLUSTERED ([Contact])
go

ALTER TABLE [Users] ADD CONSTRAINT [EmailID] UNIQUE CLUSTERED ([EmailID])
go

ALTER TABLE [Users] ADD CONSTRAINT [UserID] UNIQUE CLUSTERED ([UserID])
go

ALTER TABLE [Users] ADD CONSTRAINT [UserBankID] UNIQUE CLUSTERED ([UserBankID])
go

-- Table Invoices

```

```

CREATE TABLE [Invoices]
(
    [InvoiceID] Varchar(20) NOT NULL,
    [UserID] Varchar(10) NOT NULL,
    [PaymentType] Varchar(5) NOT NULL,
    [PaymentDate] Date NOT NULL,
    [PaymentAmount] Money NOT NULL,
    [PaymentMode] Varchar(10) NOT NULL
)
go

-- Create indexes for table Invoices

CREATE INDEX [IX_Relationship4] ON [Invoices] ([UserID])
go

-- Add keys for table Invoices

ALTER TABLE [Invoices] ADD CONSTRAINT [Key6] PRIMARY KEY ([InvoiceID])
go

ALTER TABLE [Invoices] ADD CONSTRAINT [InvoiceID] UNIQUE CLUSTERED ([InvoiceID])
go

ALTER TABLE [Invoices] ADD CONSTRAINT [UserID] UNIQUE CLUSTERED ([UserID])
go

-- Table Downloads

CREATE TABLE [Downloads]
(
    [DownloadID] Varchar(10) NOT NULL,
    [AppID] Varchar(10) NOT NULL,
    [UserID] Varchar(10) NOT NULL,
    [DownloadDate] Date NOT NULL,
    [VersionName] Varchar(10) NOT NULL
)
go

-- Create indexes for table Downloads

CREATE INDEX [IX_Relationship9] ON [Downloads] ([UserID])
go

CREATE INDEX [IX_Relationship13] ON [Downloads] ([AppID])
go

CREATE INDEX [IX_Relationship13] ON [Downloads] ([UserID])
go

-- Add keys for table Downloads

ALTER TABLE [Downloads] ADD CONSTRAINT [Key7] PRIMARY KEY ([DownloadID])
go

ALTER TABLE [Downloads] ADD CONSTRAINT [AppID] UNIQUE CLUSTERED ([AppID])
go

```

```

ALTER TABLE [Downloads] ADD CONSTRAINT [DownloadID] UNIQUE CLUSTERED ([DownloadID])
go

ALTER TABLE [Downloads] ADD CONSTRAINT [UserID] UNIQUE CLUSTERED ([UserID])
go

-- Table Version

CREATE TABLE [Version]
(
    [VersionID] Varchar(10) NOT NULL,
    [VersionName] Varchar(10) NOT NULL,
    [ReleaseDate] Date NOT NULL,
    [AppID] Varchar(10) NOT NULL
)
go

-- Create indexes for table Version

CREATE INDEX [IX_Relationship5] ON [Version] ([AppID])
go

-- Add keys for table Version

ALTER TABLE [Version] ADD CONSTRAINT [Key8] PRIMARY KEY ([VersionID])
go

ALTER TABLE [Version] ADD CONSTRAINT [AppID] UNIQUE CLUSTERED ([AppID])
go

ALTER TABLE [Version] ADD CONSTRAINT [VersionID] UNIQUE CLUSTERED ([VersionID])
go

-- Table Devices

CREATE TABLE [Devices]
(
    [DeviceID] Varchar(10) NOT NULL,
    [DeviceName] Varchar(20) NOT NULL,
    [DeviceType] Varchar(20) NOT NULL,
    [UserID] Varchar(10) NOT NULL
)
go

-- Create indexes for table Devices

CREATE INDEX [IX_Relationship15] ON [Devices] ([UserID])
go

-- Add keys for table Devices

ALTER TABLE [Devices] ADD CONSTRAINT [Key9] PRIMARY KEY ([DeviceID])
go

ALTER TABLE [Devices] ADD CONSTRAINT [DeviceID] UNIQUE CLUSTERED ([DeviceID])
go

```



```

ALTER TABLE [Devices] ADD CONSTRAINT [DeviceName] UNIQUE CLUSTERED ([DeviceName])
go

ALTER TABLE [Devices] ADD CONSTRAINT [UserID] UNIQUE CLUSTERED ([UserID])
go

-- Table DeveloperAppBridge

CREATE TABLE [DeveloperAppBridge]
(
    [AppID] Varchar(10) NOT NULL,
    [DeveloperID] Varchar(10) NOT NULL
)
go

-- Add keys for table DeveloperAppBridge

ALTER TABLE [DeveloperAppBridge] ADD CONSTRAINT [Key16] PRIMARY KEY ([AppID],[DeveloperID])
go

-- Table ApplicationUserBridge

CREATE TABLE [ApplicationUserBridge]
(
    [AppID] Varchar(10) NOT NULL,
    [UserID] Varchar(10) NOT NULL
)
go

-- Add keys for table ApplicationUserBridge

ALTER TABLE [ApplicationUserBridge] ADD CONSTRAINT [Key15] PRIMARY KEY ([AppID],[UserID])
go

-- Table DevBankDetails

CREATE TABLE [DevBankDetails]
(
    [DevBankID] Varchar(10) NOT NULL,
    [DevAddress] Varchar(50) NOT NULL,
    [DevRoutingNo] Int NOT NULL,
    [DeveloperAccNo] Int NOT NULL
)
go

-- Add keys for table DevBankDetails

ALTER TABLE [DevBankDetails] ADD CONSTRAINT [Key14] PRIMARY KEY ([DevBankID])
go

ALTER TABLE [DevBankDetails] ADD CONSTRAINT [DevBankID] UNIQUE CLUSTERED ([DevBankID])
go

ALTER TABLE [DevBankDetails] ADD CONSTRAINT [DevAddress] UNIQUE CLUSTERED ([DevAddress])
go

ALTER TABLE [DevBankDetails] ADD CONSTRAINT [DeveloperAccNo] UNIQUE CLUSTERED ([DeveloperAccNo])
go

```

-- Table UserBankDetails

```
CREATE TABLE [UserBankDetails]
(
    [UserBankID] Varchar(10) NOT NULL,
    [UserBankName1] Varchar(20) NOT NULL,
    [UserBankName] Int NOT NULL,
    [UserBankRoutingno] Int NOT NULL,
    [UserCardNo] Int NOT NULL,
    [CardExpDate] Date NOT NULL
)
go
```

-- Add keys for table UserBankDetails

```
ALTER TABLE [UserBankDetails] ADD CONSTRAINT [Key13] PRIMARY KEY ([UserBankID])
go
```

```
ALTER TABLE [UserBankDetails] ADD CONSTRAINT [UserCardNo] UNIQUE CLUSTERED ([UserCardNo])
go
```

```
ALTER TABLE [UserBankDetails] ADD CONSTRAINT [UserBankID] UNIQUE CLUSTERED ([UserBankID])
go
```

-- Table CategoryApplicationBridge

```
CREATE TABLE [CategoryApplicationBridge]
(
    [CategoryID] Varchar(10) NOT NULL,
    [AppID] Varchar(10) NOT NULL
)
go
```

-- Add keys for table CategoryApplicationBridge

```
ALTER TABLE [CategoryApplicationBridge] ADD CONSTRAINT [Key17] PRIMARY KEY
([CategoryID],[AppID])
go
```

-- Create foreign keys (relationships) section -----

```
ALTER TABLE [Invoices] ADD CONSTRAINT [One to Many] FOREIGN KEY ([UserID]) REFERENCES [Users]
([UserID]) ON UPDATE NO ACTION ON DELETE NO ACTION
go
```

```
ALTER TABLE [Ratings] ADD CONSTRAINT [One to Many] FOREIGN KEY ([UserID]) REFERENCES [Users]
([UserID]) ON UPDATE NO ACTION ON DELETE NO ACTION
go
```

```
ALTER TABLE [Version] ADD CONSTRAINT [One to Many] FOREIGN KEY ([AppID]) REFERENCES
[Application] ([AppID]) ON UPDATE NO ACTION ON DELETE NO ACTION
go
```

```
ALTER TABLE [ApplicationUserBridge] ADD CONSTRAINT [Many to Many] FOREIGN KEY ([AppID])  
REFERENCES [Application] ([AppID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [ApplicationUserBridge] ADD CONSTRAINT [Many to Many] FOREIGN KEY ([UserID])  
REFERENCES [Users] ([UserID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [DeveloperAppBridge] ADD CONSTRAINT [Many to Many] FOREIGN KEY ([AppID]) REFERENCES  
[Application] ([AppID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [DeveloperAppBridge] ADD CONSTRAINT [Many to Many] FOREIGN KEY ([DeveloperID])  
REFERENCES [Developer] ([DeveloperID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [Ratings] ADD CONSTRAINT [One to Many4] FOREIGN KEY ([AppID]) REFERENCES  
[Application] ([AppID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [Devices] ADD CONSTRAINT [One to Many] FOREIGN KEY ([UserID]) REFERENCES [Users]  
([UserID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [Developer] ADD CONSTRAINT [One to One] FOREIGN KEY ([DevBankID]) REFERENCES  
[DevBankDetails] ([DevBankID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [Downloads] ADD CONSTRAINT [One to Many] FOREIGN KEY ([AppID]) REFERENCES  
[Application] ([AppID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

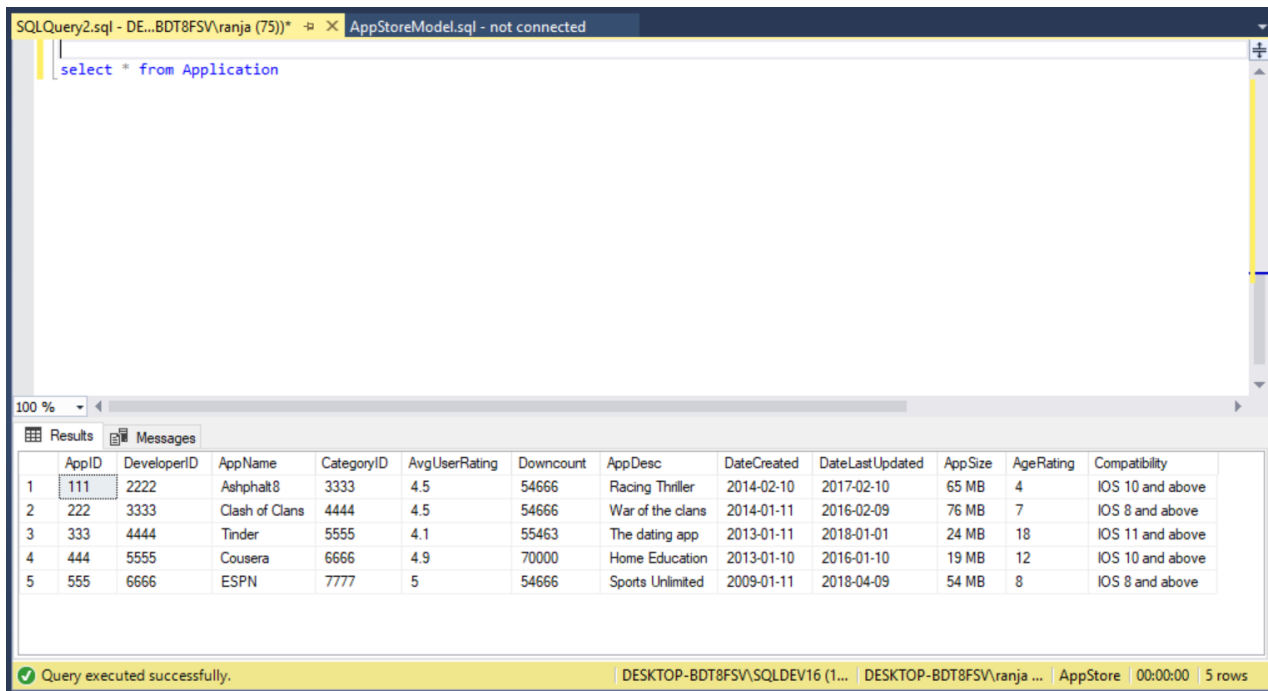
```
ALTER TABLE [Downloads] ADD CONSTRAINT [One to Many] FOREIGN KEY ([UserID]) REFERENCES [Devices]  
([UserID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [Users] ADD CONSTRAINT [One to One] FOREIGN KEY ([UserBankID]) REFERENCES  
[UserBankDetails] ([UserBankID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [CategoryApplicationBridge] ADD CONSTRAINT [Many to Many] FOREIGN KEY ([CategoryID])  
REFERENCES [Category] ([CategoryID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

```
ALTER TABLE [CategoryApplicationBridge] ADD CONSTRAINT [Many to Many] FOREIGN KEY ([AppID])  
REFERENCES [Application] ([AppID]) ON UPDATE NO ACTION ON DELETE NO ACTION  
go
```

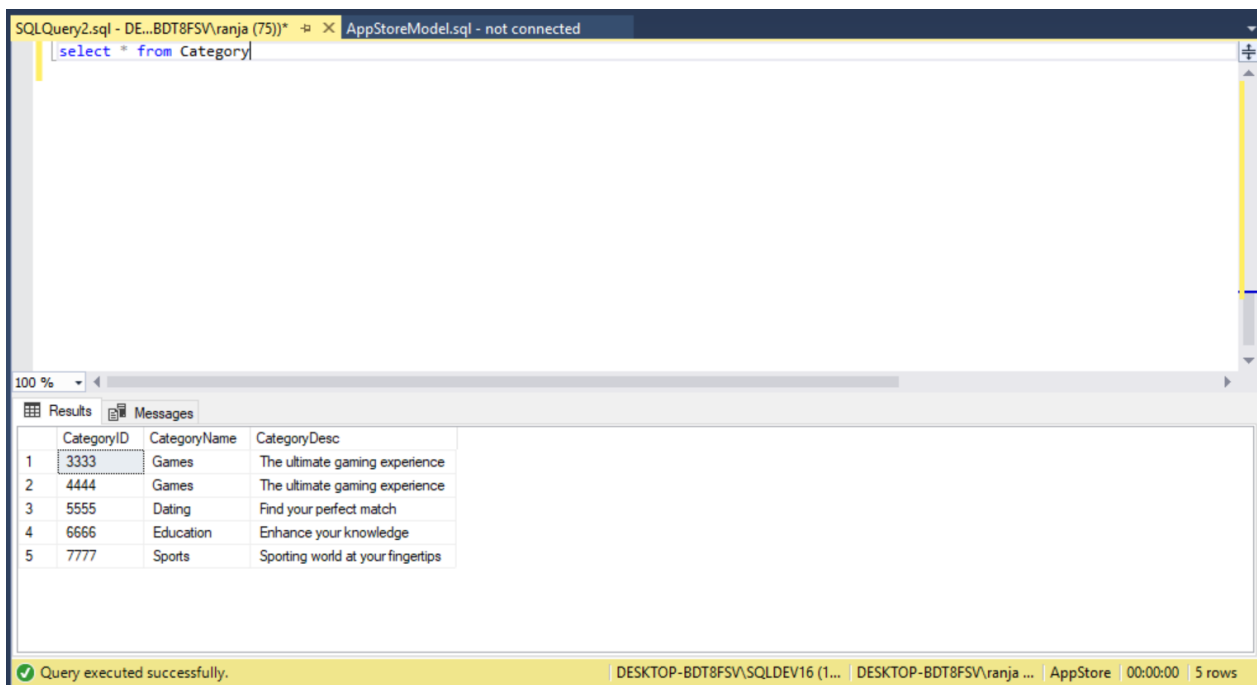
## Entity Applications



The screenshot shows a SQL Server Enterprise Manager window with a query executed against the 'AppStoreModel.sql' database. The query is 'select \* from Application'. The results are displayed in a table with 13 columns: AppID, DeveloperID, AppName, CategoryID, AvgUserRating, DownloadCount, AppDesc, DateCreated, DateLastUpdated, AppSize, AgeRating, and Compatibility. The status bar at the bottom indicates 'Query executed successfully.' and '5 rows'.

	AppID	DeveloperID	AppName	CategoryID	AvgUserRating	DownloadCount	AppDesc	DateCreated	DateLastUpdated	AppSize	AgeRating	Compatibility
1	111	2222	Ashphalt8	3333	4.5	54666	Racing Thriller	2014-02-10	2017-02-10	65 MB	4	IOS 10 and above
2	222	3333	Clash of Clans	4444	4.5	54666	War of the clans	2014-01-11	2016-02-09	76 MB	7	IOS 8 and above
3	333	4444	Tinder	5555	4.1	55463	The dating app	2013-01-11	2018-01-01	24 MB	18	IOS 11 and above
4	444	5555	Cousera	6666	4.9	70000	Home Education	2013-01-10	2016-01-10	19 MB	12	IOS 10 and above
5	555	6666	ESPN	7777	5	54666	Sports Unlimited	2009-01-11	2018-04-09	54 MB	8	IOS 8 and above

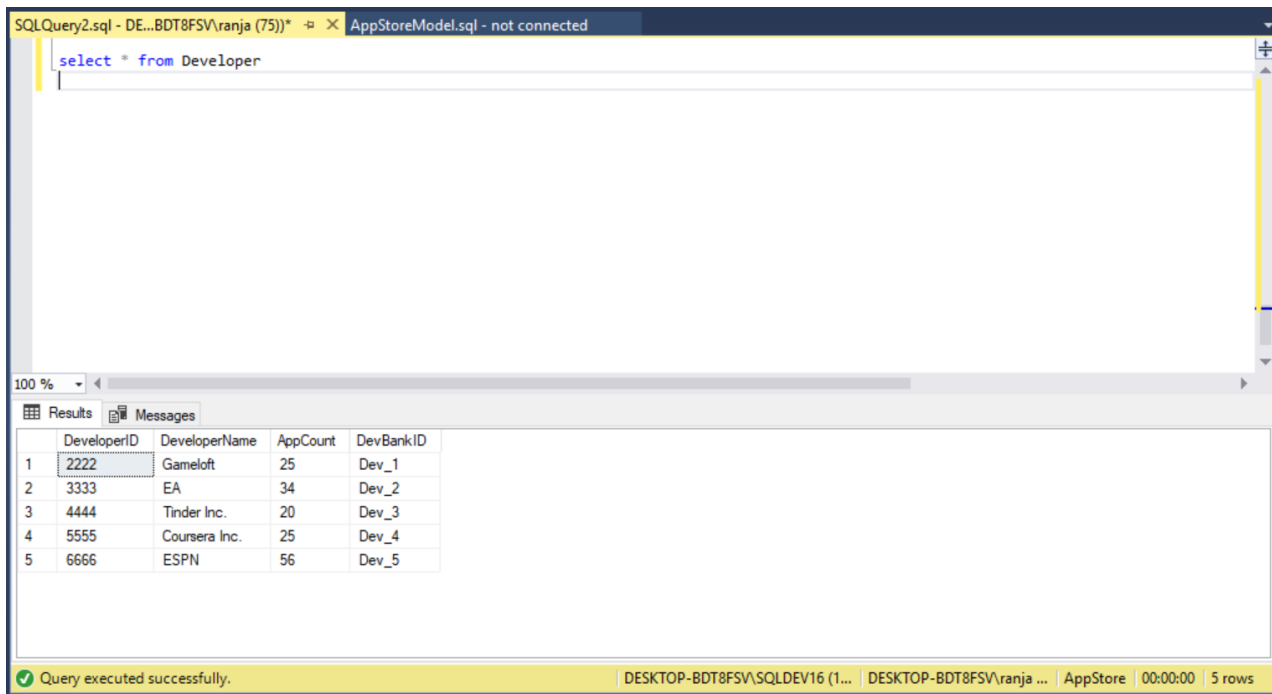
## Entity Category



The screenshot shows a SQL Server Enterprise Manager window with a query executed against the 'AppStoreModel.sql' database. The query is 'select \* from Category'. The results are displayed in a table with 3 columns: CategoryID, CategoryName, and CategoryDesc. The status bar at the bottom indicates 'Query executed successfully.' and '5 rows'.

	CategoryID	CategoryName	CategoryDesc
1	3333	Games	The ultimate gaming experience
2	4444	Games	The ultimate gaming experience
3	5555	Dating	Find your perfect match
4	6666	Education	Enhance your knowledge
5	7777	Sports	Sporting world at your fingertips

## Entity Developer

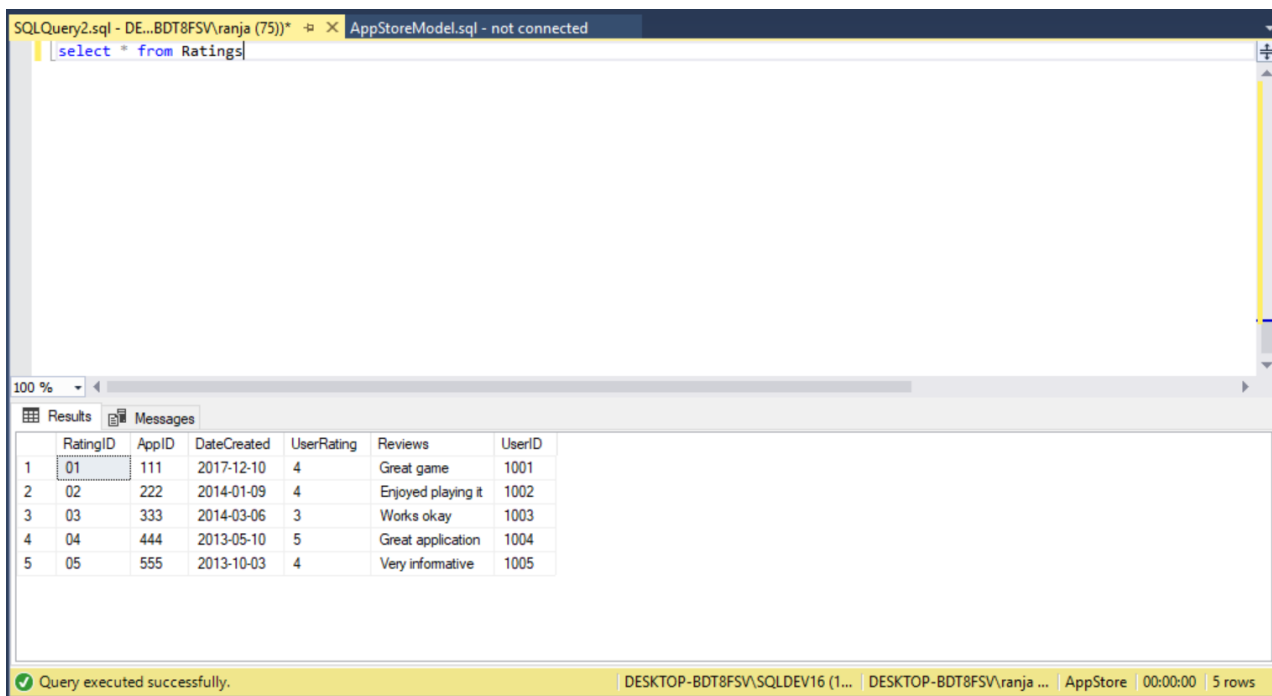


The screenshot shows a SQL Query Editor window with the following tabs: "SQLQuery2.sql - DE...BDT8FSV\ranja (75))\*" and "AppStoreModel.sql - not connected". The query entered is `select * from Developer`. The results pane shows a table with 5 rows and 4 columns: DeveloperID, DeveloperName, AppCount, and DevBankID.

	DeveloperID	DeveloperName	AppCount	DevBankID
1	2222	Gameloft	25	Dev_1
2	3333	EA	34	Dev_2
3	4444	Tinder Inc.	20	Dev_3
4	5555	Coursera Inc.	25	Dev_4
5	6666	ESPN	56	Dev_5

Query executed successfully. DESKTOP-BDT8FSV\SQLDEV16 (1... DESKTOP-BDT8FSV\ranja ... AppStore 00:00:00 5 rows

## Entity Ratings

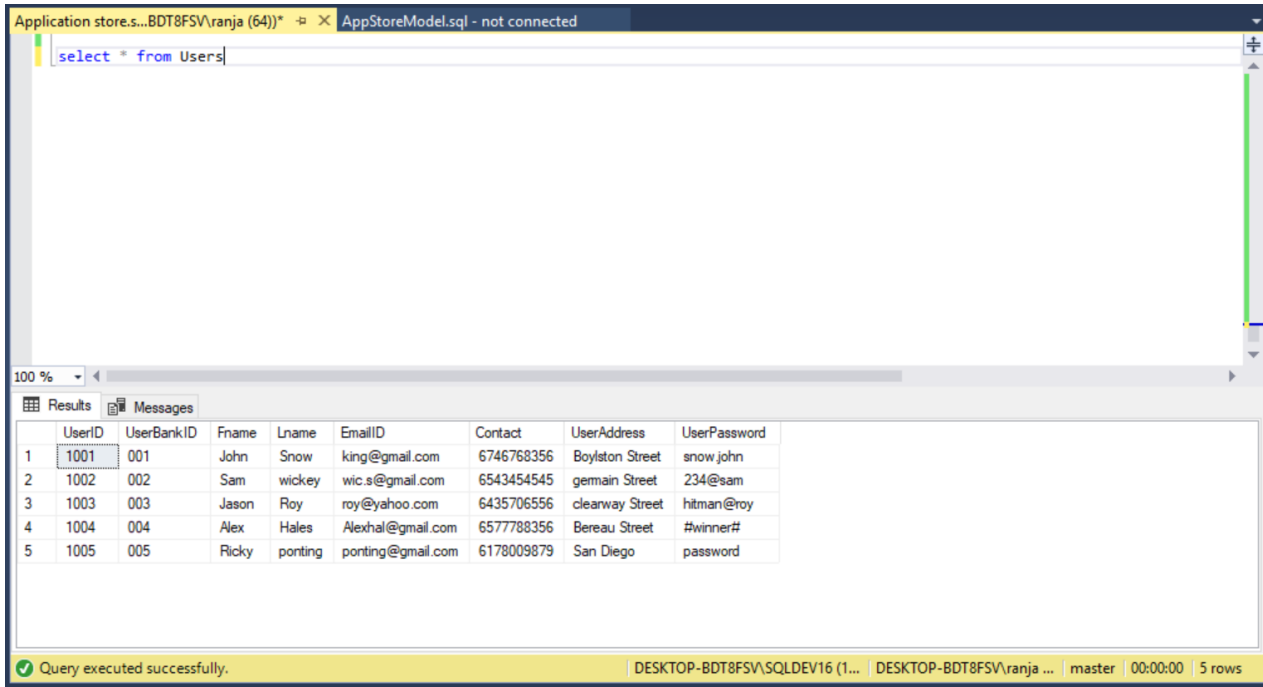


The screenshot shows a SQL Query Editor window with the following tabs: "SQLQuery2.sql - DE...BDT8FSV\ranja (75))\*" and "AppStoreModel.sql - not connected". The query entered is `select * from Ratings`. The results pane shows a table with 5 rows and 7 columns: RatingID, AppID, DateCreated, UserRating, Reviews, and UserID.

	RatingID	AppID	DateCreated	UserRating	Reviews	UserID
1	01	111	2017-12-10	4	Great game	1001
2	02	222	2014-01-09	4	Enjoyed playing it	1002
3	03	333	2014-03-06	3	Works okay	1003
4	04	444	2013-05-10	5	Great application	1004
5	05	555	2013-10-03	4	Very informative	1005

Query executed successfully. DESKTOP-BDT8FSV\SQLDEV16 (1... DESKTOP-BDT8FSV\ranja ... AppStore 00:00:00 5 rows

## Entity Users



Application store.s...BDT8FSV\ranja (64)) \* X AppStoreModel.sql - not connected

```
select * from Users
```

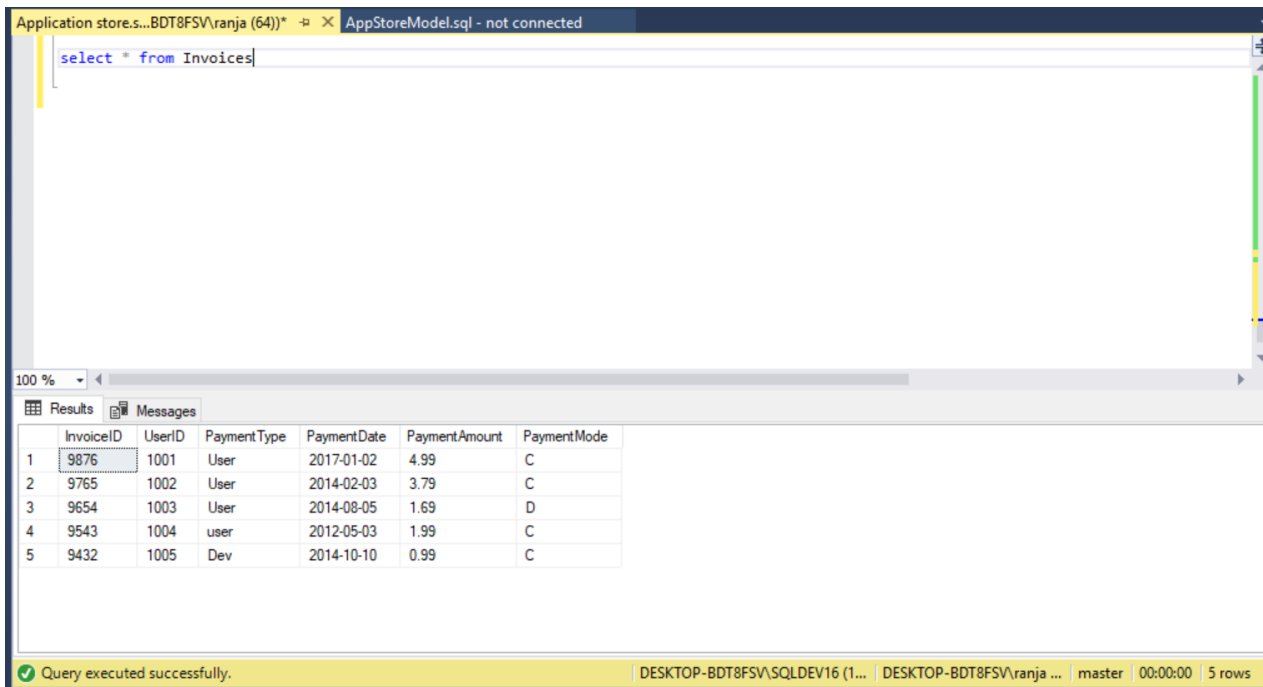
100 %

Results Messages

	UserID	UserBankID	Fname	Lname	EmailID	Contact	UserAddress	UserPassword
1	1001	001	John	Snow	king@gmail.com	6746768356	Boylston Street	snow john
2	1002	002	Sam	wickey	wic.s@gmail.com	6543454545	gemain Street	234@sam
3	1003	003	Jason	Roy	roy@yahoo.com	6435706556	clearway Street	hitman@roy
4	1004	004	Alex	Hales	Alexhal@gmail.com	6577788356	Bereau Street	#winner#
5	1005	005	Ricky	ponting	ponting@gmail.com	6178009879	San Diego	password

Query executed successfully. DESKTOP-BDT8FSV\SQLDEV16 (1... DESKTOP-BDT8FSV\ranja ... master 00:00:00 5 rows

## Entity Invoices



Application store.s...BDT8FSV\ranja (64)) \* X AppStoreModel.sql - not connected

```
select * from Invoices
```

100 %

Results Messages

	InvoiceID	UserID	PaymentType	PaymentDate	PaymentAmount	PaymentMode
1	9876	1001	User	2017-01-02	4.99	C
2	9765	1002	User	2014-02-03	3.79	C
3	9654	1003	User	2014-08-05	1.69	D
4	9543	1004	user	2012-05-03	1.99	C
5	9432	1005	Dev	2014-10-10	0.99	C

Query executed successfully. DESKTOP-BDT8FSV\SQLDEV16 (1... DESKTOP-BDT8FSV\ranja ... master 00:00:00 5 rows

## Entity Downloads

The screenshot shows a SQL Server Enterprise Manager window with the query `select * from Downloads` executed. The results are displayed in a table with 5 rows. The status bar at the bottom indicates "Query executed successfully." and "5 rows".

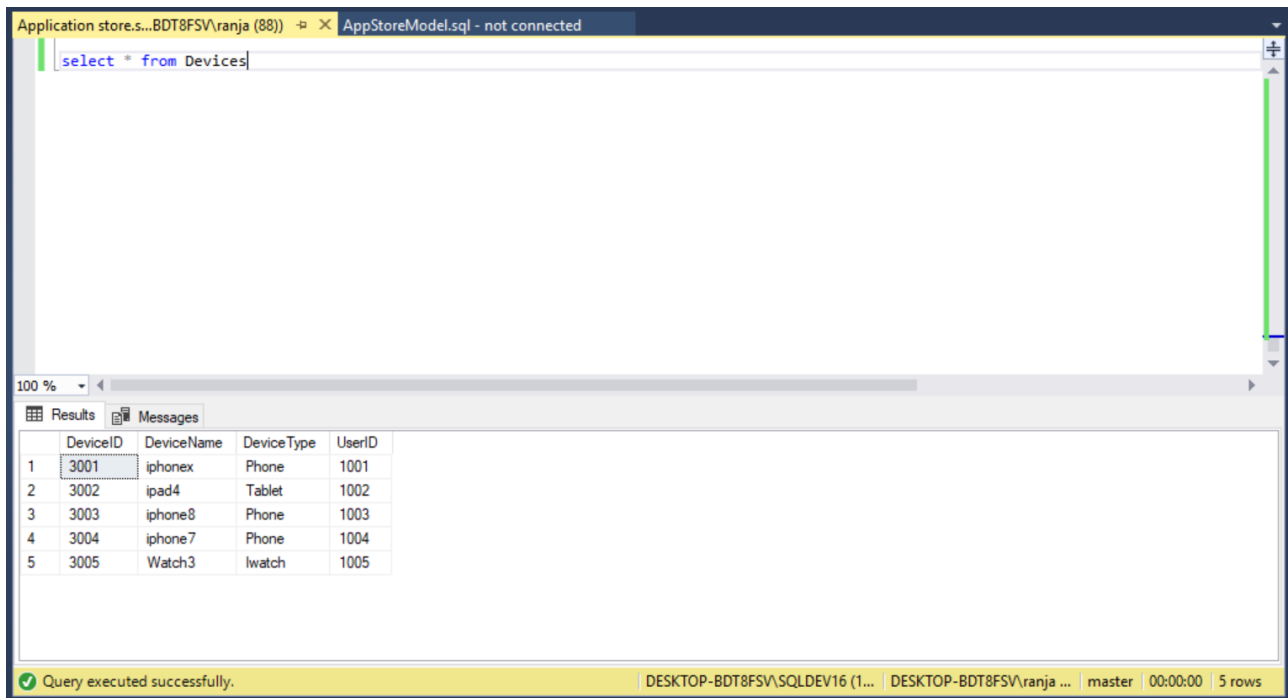
	DownloadID	AppID	UserID	DownloadDate	VersionName
1	2001	111	1001	2018-01-01	1.2
2	2002	222	1002	2013-01-02	1.3
3	2003	333	1003	2014-01-03	2.1
4	2004	444	1004	2001-02-05	2.4
5	2005	555	1005	2004-01-05	1.5

## Entity Version

The screenshot shows a SQL Server Enterprise Manager window with the query `select * from Version` executed. The results are displayed in a table with 5 rows. The status bar at the bottom indicates "Query executed successfully." and "5 rows".

	VersionID	VersionName	ReleaseDate	AppID
1	123	1.2	2015-03-09	111
2	134	1.3	2016-03-04	222
3	145	2.1	2005-12-10	333
4	156	2.4	2017-03-10	444
5	167	1.5	2015-12-10	555

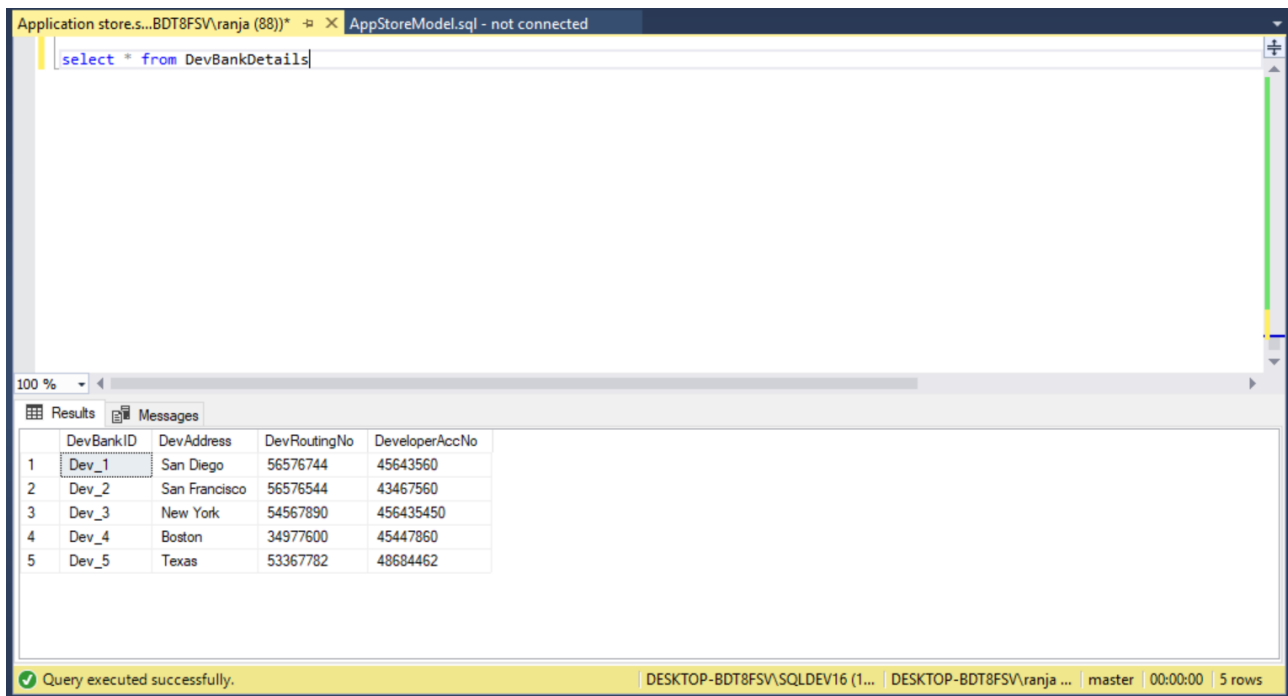
## Entity Devices



The screenshot shows a SQL Server Enterprise Manager window with the query `select * from Devices` executed. The results are displayed in a table with 5 rows and 4 columns: DeviceID, DeviceName, DeviceType, and UserID. The status bar at the bottom indicates the query was executed successfully on the 'master' database, returning 5 rows in 00:00:00.

	DeviceID	DeviceName	DeviceType	UserID
1	3001	iphonex	Phone	1001
2	3002	ipad4	Tablet	1002
3	3003	iphone8	Phone	1003
4	3004	iphone7	Phone	1004
5	3005	Watch3	lwatch	1005

## Entity DevBankDetails



The screenshot shows a SQL Server Enterprise Manager window with the query `select * from DevBankDetails` executed. The results are displayed in a table with 5 rows and 4 columns: DevBankID, DevAddress, DevRoutingNo, and DeveloperAccNo. The status bar at the bottom indicates the query was executed successfully on the 'master' database, returning 5 rows in 00:00:00.

	DevBankID	DevAddress	DevRoutingNo	DeveloperAccNo
1	Dev_1	San Diego	56576744	45643560
2	Dev_2	San Francisco	56576544	43467560
3	Dev_3	New York	54567890	456435450
4	Dev_4	Boston	34977600	45447860
5	Dev_5	Texas	53367782	48684462



## Entity UserBankDetails

Application store.s...BDT8FSV\ranja (88)) \* AppStoreModel.sql - not connected

```
select * from userbankdetails
```

	UserBankID	UserBankName	UserBankRoutingno	UserCardNo	CardExpDate
1	001	BoFA	57574788	5676432345456565	2022-08-01
2	002	BoFA	57574788	6567897678765454	2021-09-01
3	003	Santander Bank	43535678	9890000076769090	2034-07-09
4	005	ICICI bank	355525425	7544567877776565	2023-08-01
5	004	Citizens bank	465435455	7676909866664343	2020-07-01

Query executed successfully. DESKTOP-BDT8FSV\SQLDEV16 (1... DESKTOP-BDT8FSV\ranja ... master 00:00:00 5 rows

## Toad Model

