WorldCover – Mini Project 1

@Author: Ajinkya Rasam

E-mail: [rasam.ajinkya23@rutgers.edu](mailto:rasam.ajinkya23@rutgers.edu)

In this mini-project I will be designing an end-to-end solution for interaction between a mobile application and backend(database) for WorldCover company.

Data Extraction

First and foremost, thing I need to do is to take a look at the data which is available on ‘Fulcrum’ using API key shared with me.

For this I am going to use a python library for fulcrum.



Further using following API key and forms I will extract the data from fulcrum website for

* Register Payment 2015/2016 *(Paid Customers)*
* Communities 2015-2016 *(Registered and Paid Communities)*
* Trigger Locations 2015/2016 *(Paid Community Locations)*

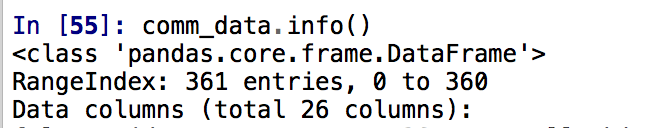
The API key is:  
**f091f608bc49672b861db74dce095437cd07974cac3da475e0d3147a280f12dd533a58c7c1da5e5f**

After which I converted it into CSV files for better understanding and using it in excel in order to get better insights.

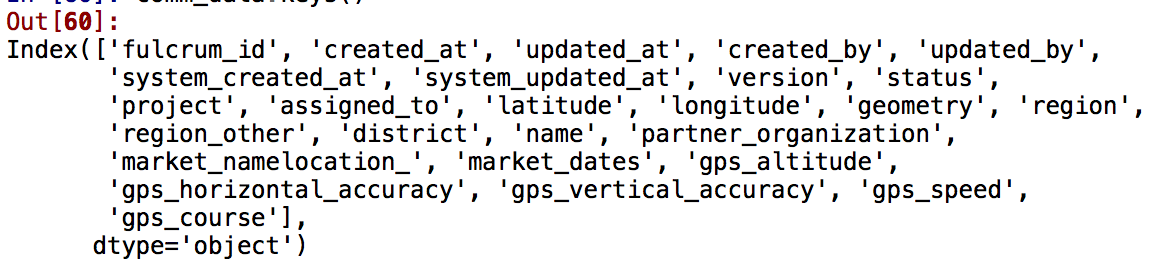
# The Data

1. Communities:



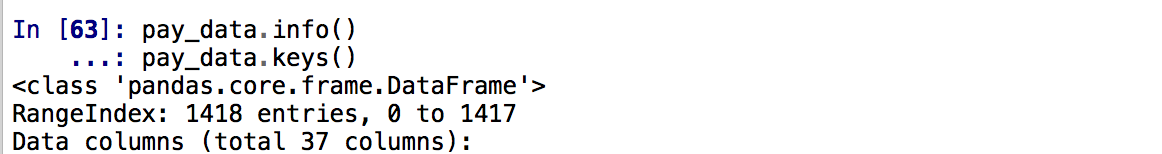
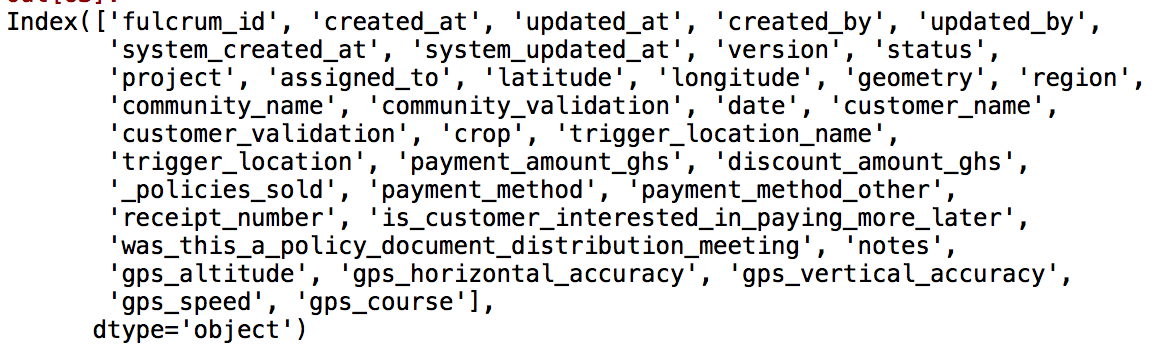


This dataframe has 362 entries and 26 attributes with a combination of integer, string and Date datatypes.



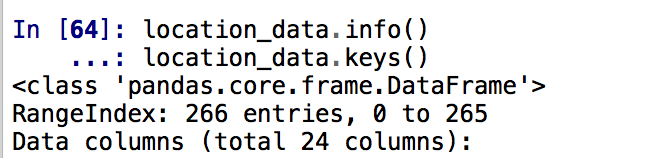
Here, ‘fulcrum\_id’ is the unique identifier for each record and therefore, is a potential candidate for primary key in our database schema.

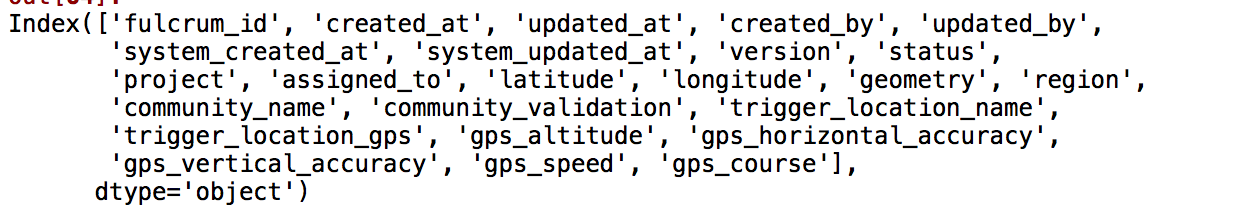
1. Payment Data:



This dataframe has 1417 entries and 37 attributes

1. Location Data:

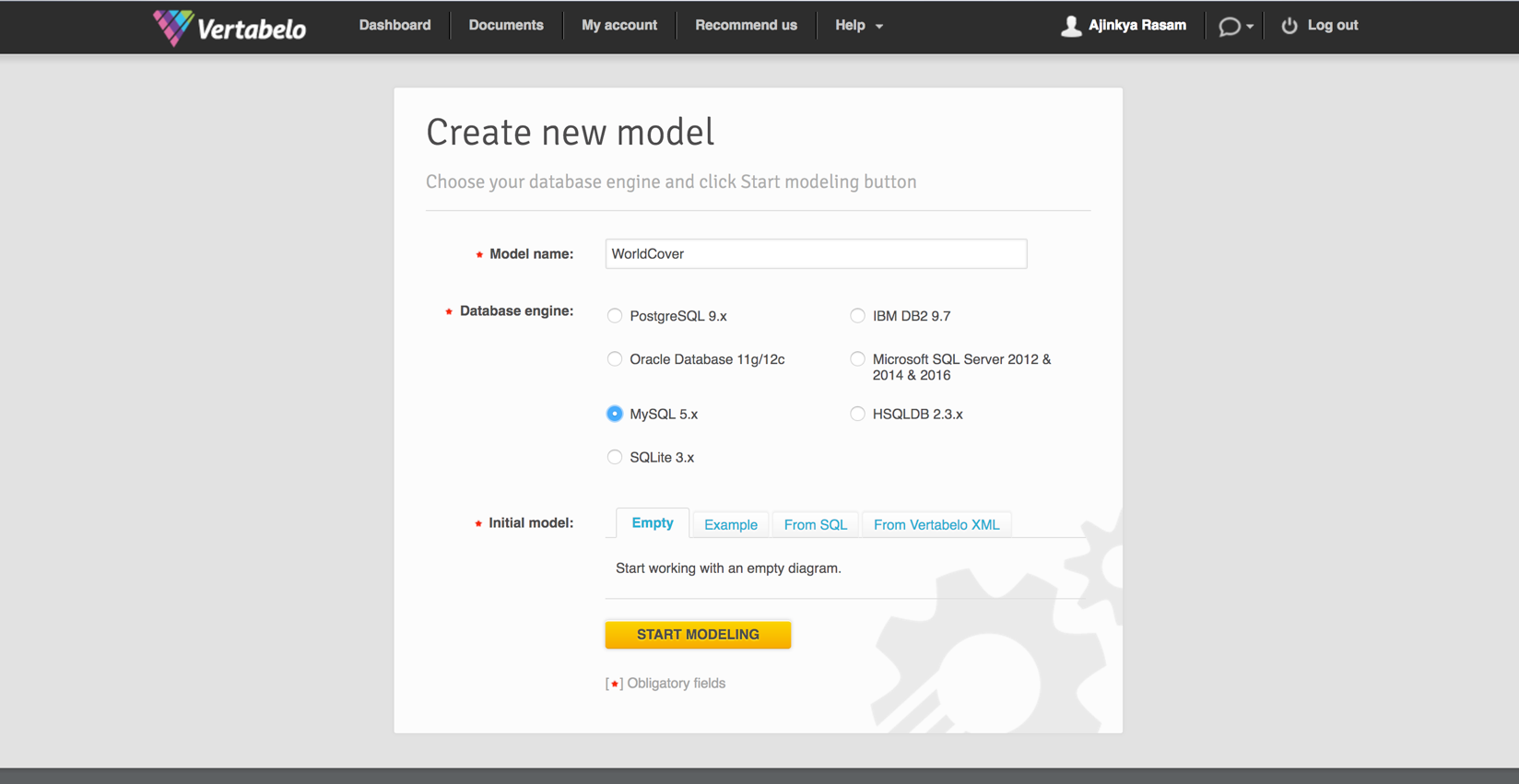




This dataframe has 266 entries and 24 attributes

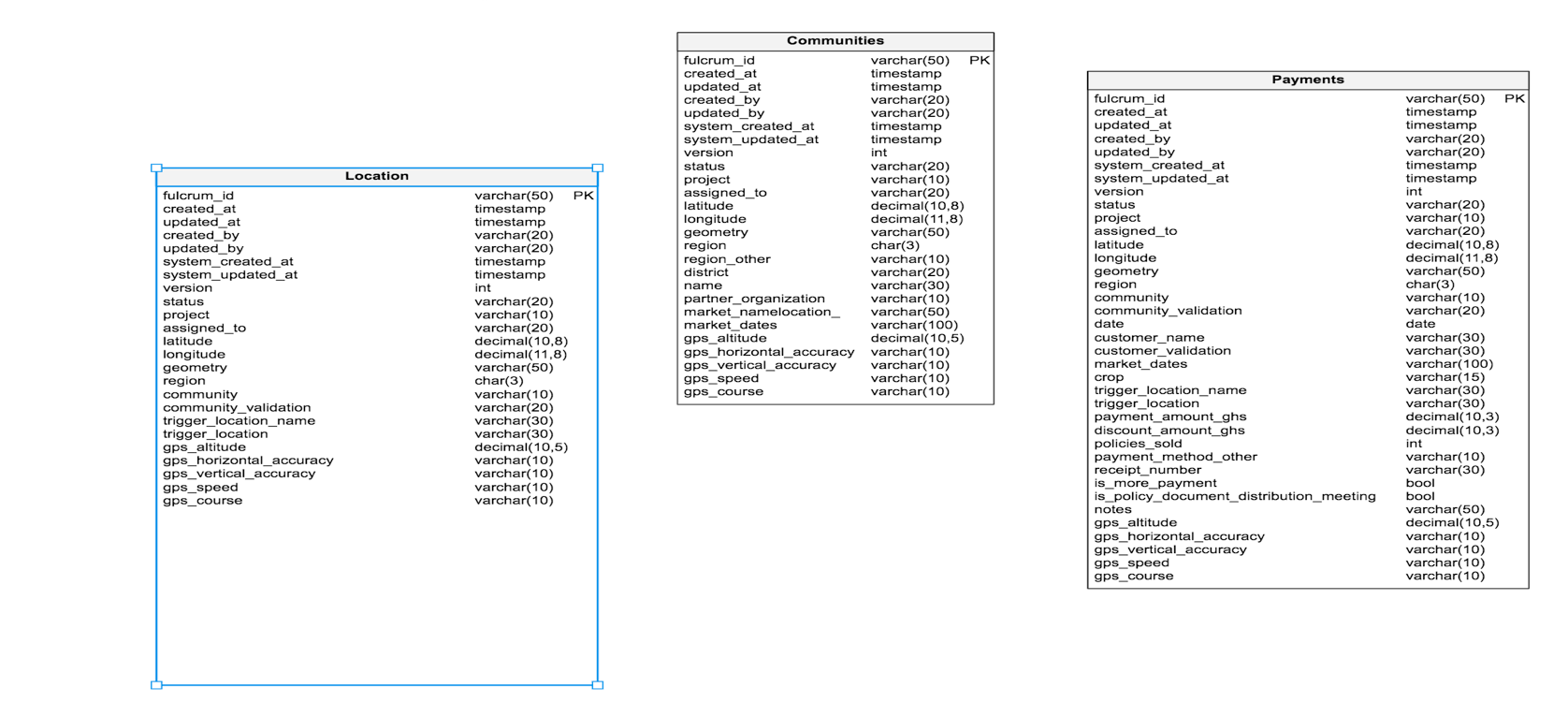
# Database Model Creation

I am going to use ‘Vertabalo’ a freeware website for model creation. It has excellent features for database creation and manipulation.



# I am planning to use MySQL 5.x model just because the data into picture right now is not huge and *MySQL* has an excellent track record over the years. Moreover, it is an open source project and thus, we get a lot of documentation in MySQL community. Having said that, we also have an option to use *PostgreSQL*.

Without normalization the data tables look like as follows:



It has a lot of redundancy and inter dependency which needs to be eliminated in order to create an efficient database design.

Attributes related to user details, location, payment, community and GPS details can be separated into different tables and these tables can further be interlinked using unique key references.

Assumptions:

* The data which is going to be inserted into DB is related to payment transactions.
* A community grows only one type of crop.
* For the columns which do not have data as of now, in future will get data which is within the scope of VARCHAR data type.

After Normalization, the database schema looks as below:

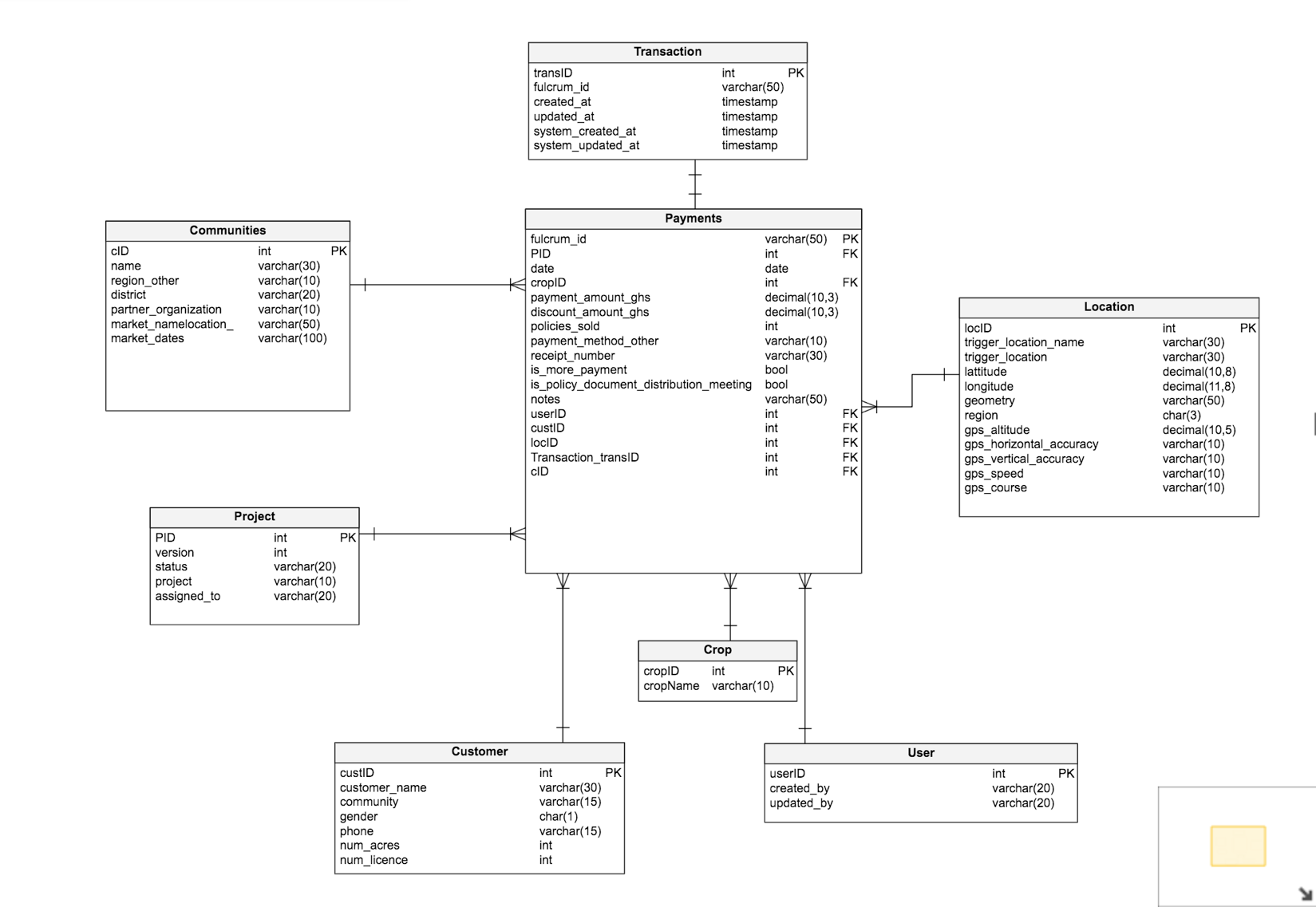


Table Info:

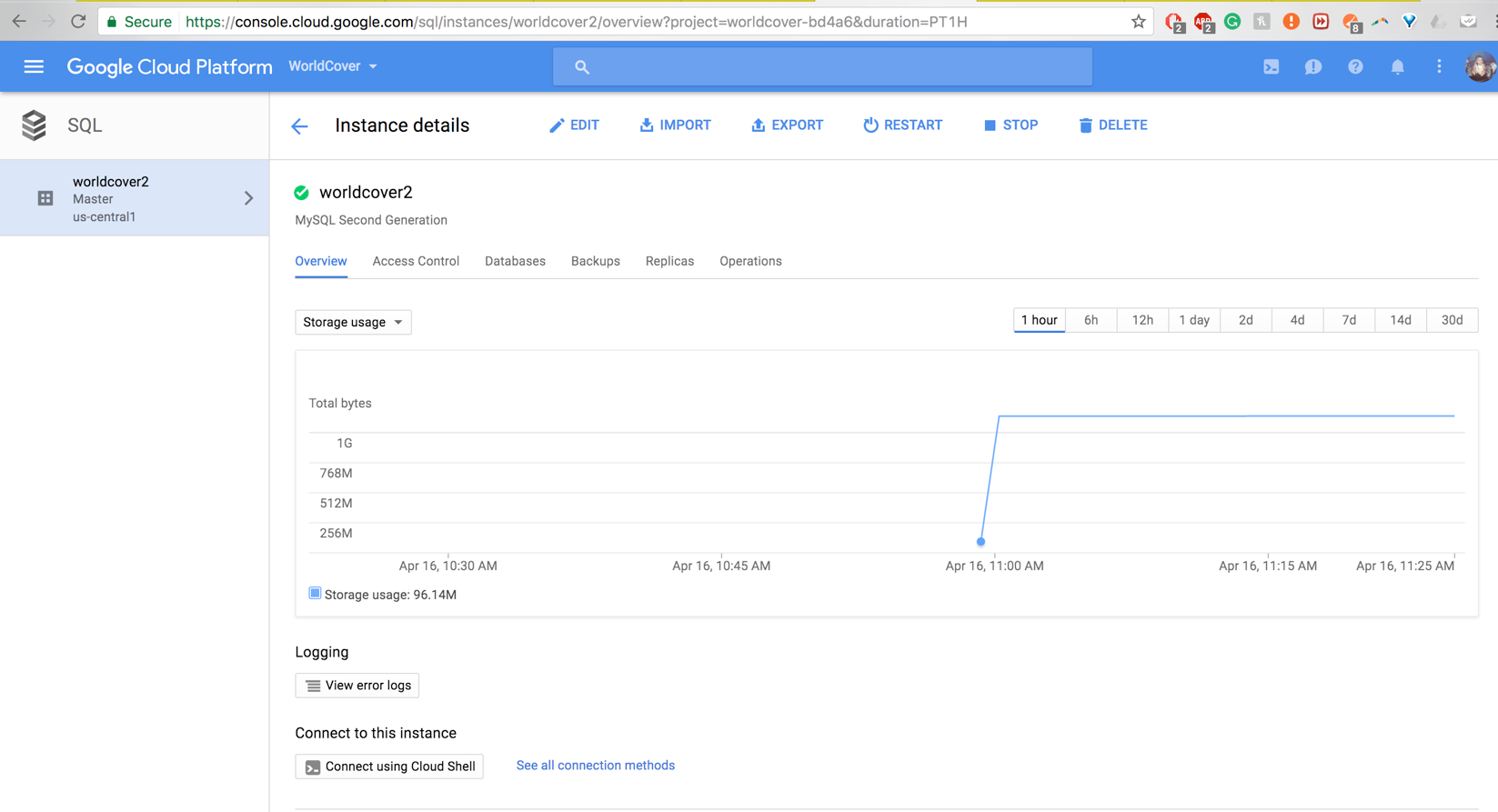
Transaction table will be utilized to log creating of upgrading of a every payment transaction. All the payment details will be stored in the ‘Payments’ table and each further attribute related to a payments transaction will have its own table. This eliminates data redundancy and saves a lot of storage space.

Mapping:

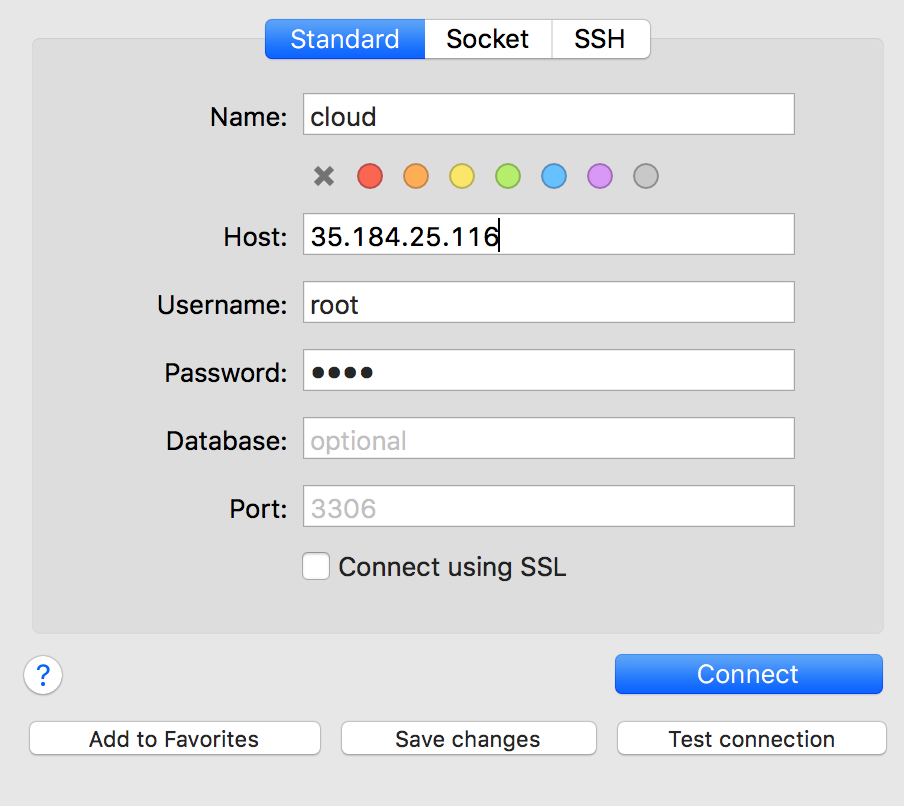
For the sake of simplicity, I have kept the field names same. These can be changed anytime in future.

Creating a cloud instance and hosting the database:

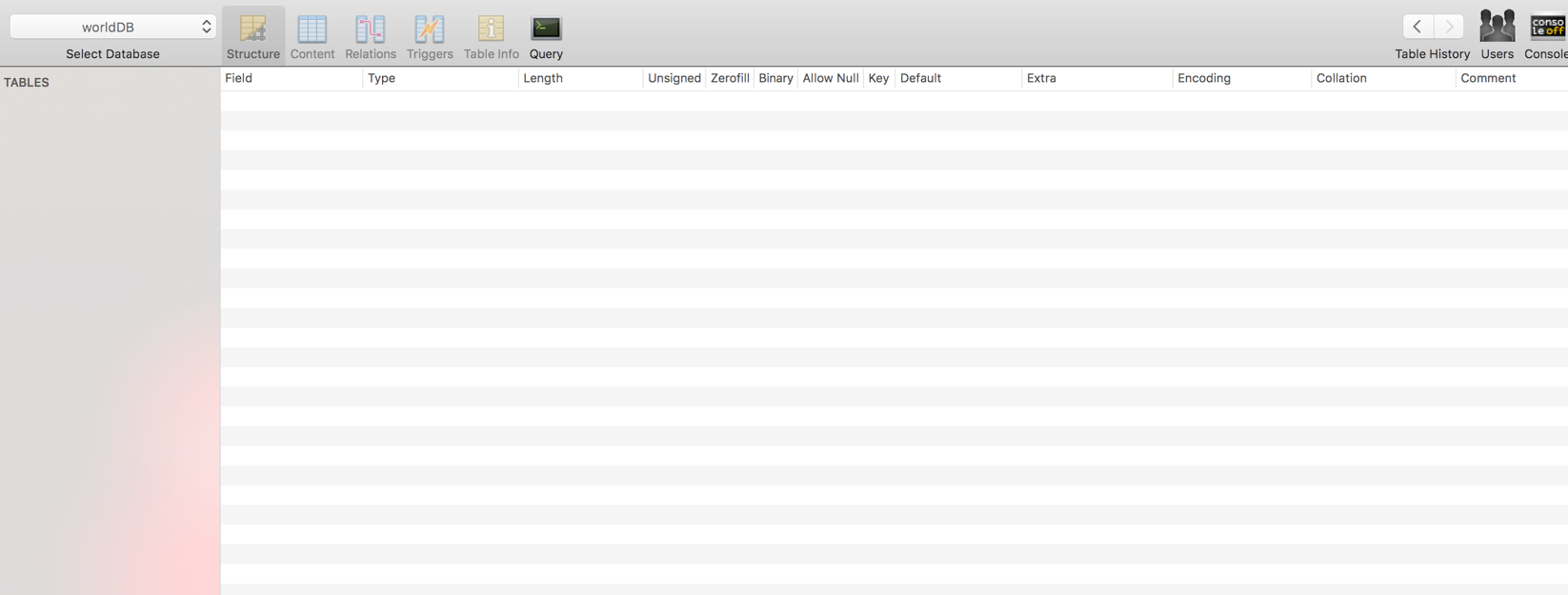
I am going to use Google Cloud for hosting my database. Spcifically, I am creating a new MySQL 5.7x instance with a 10gb of storage.



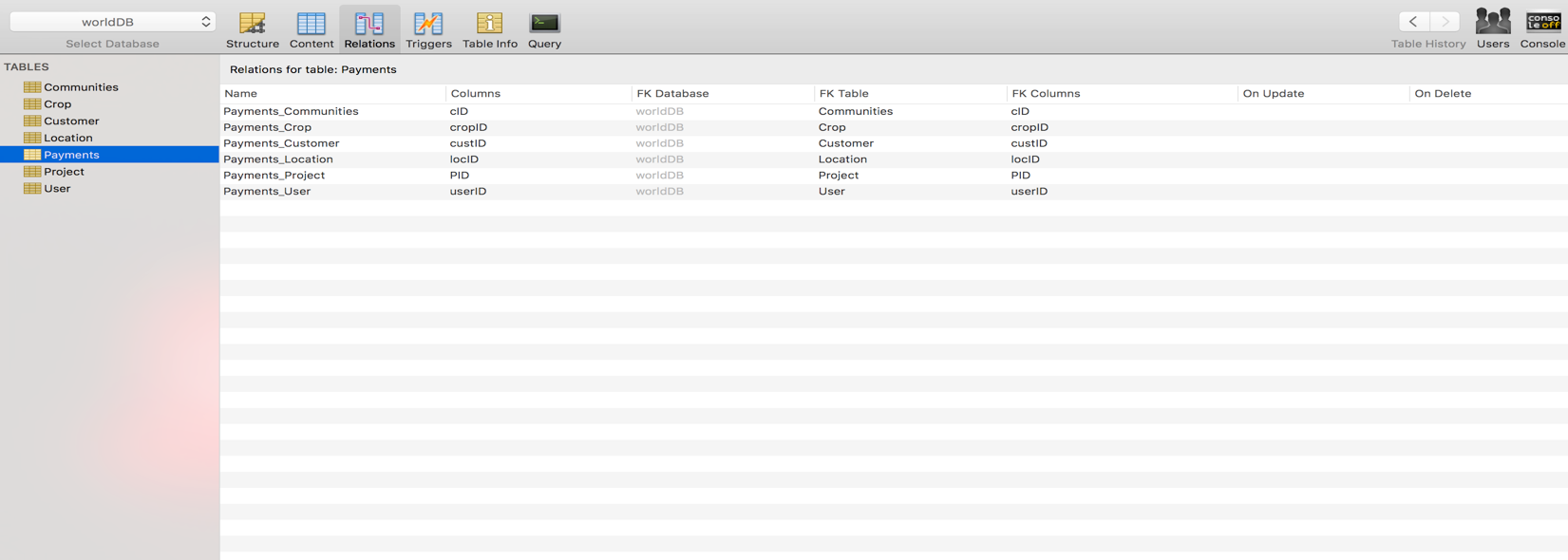
I am going to connect to this instance using MySQL client – Sequel pro.

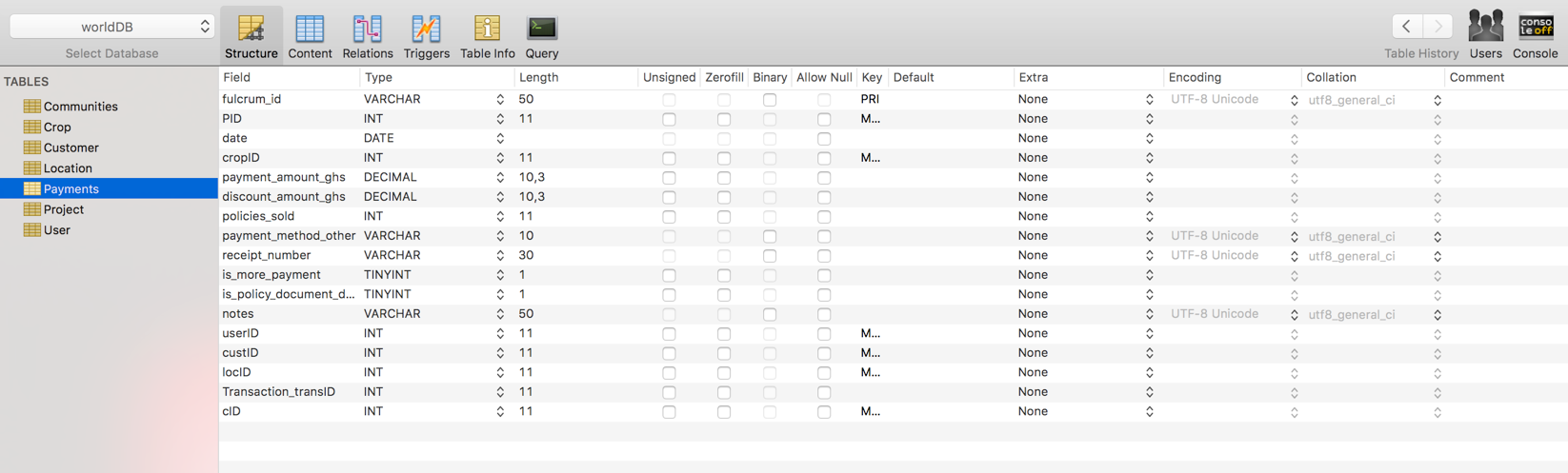


Right now there are no tables in the database. We need to right a script for creating the database schema.

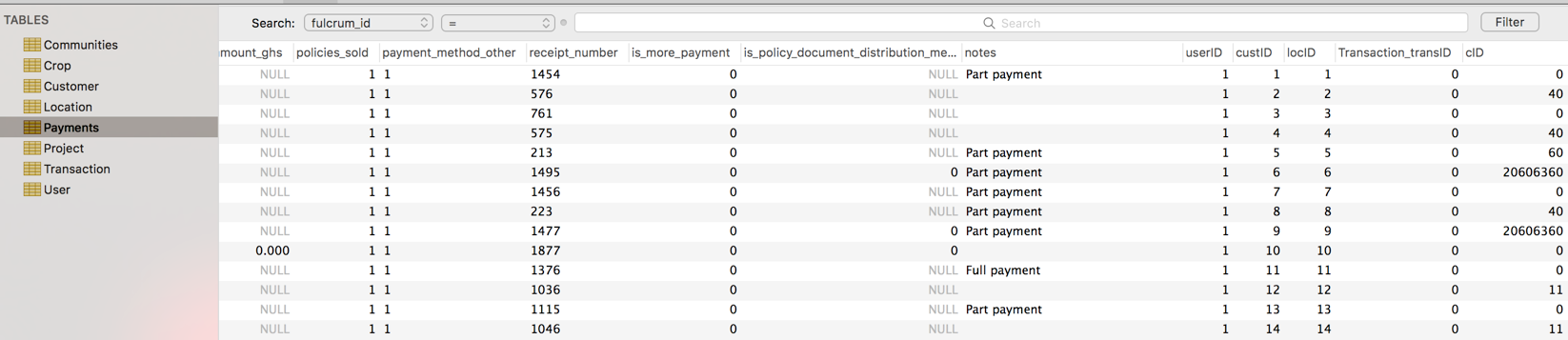


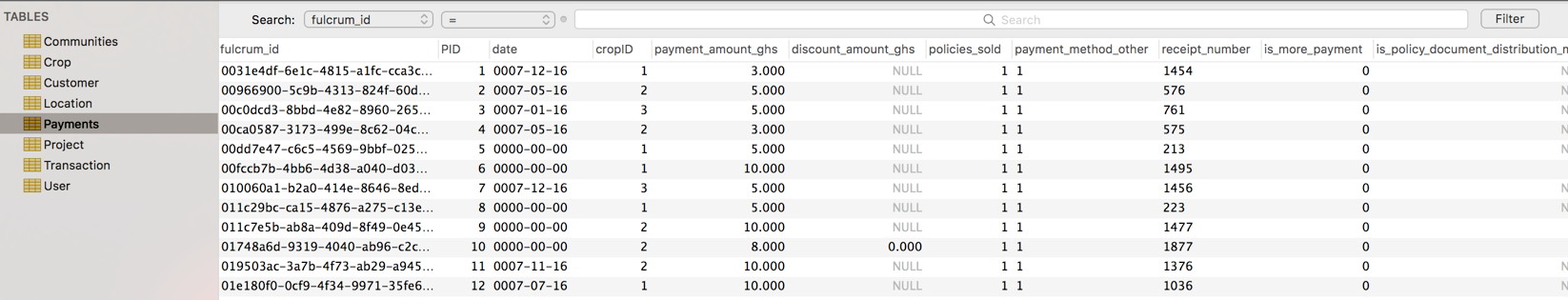
After firing the script, the database schema looks like below:

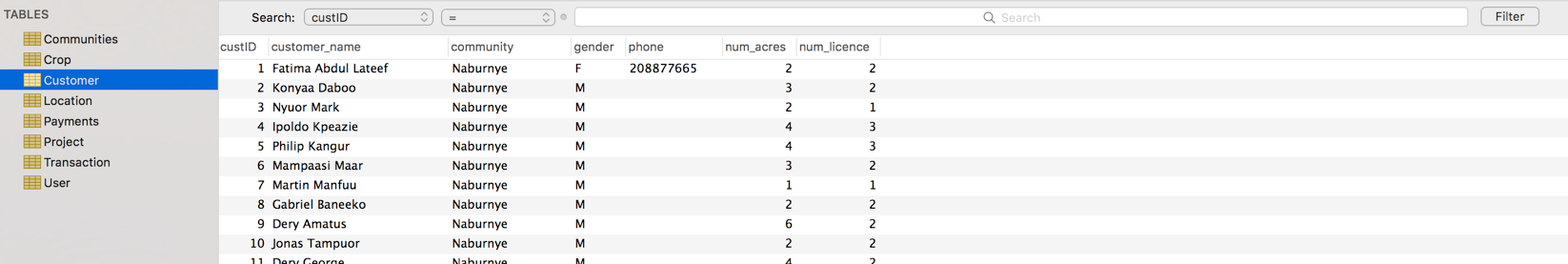


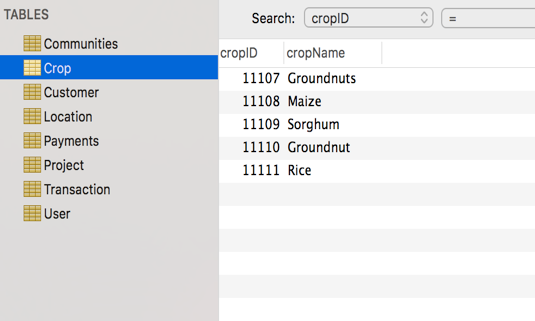


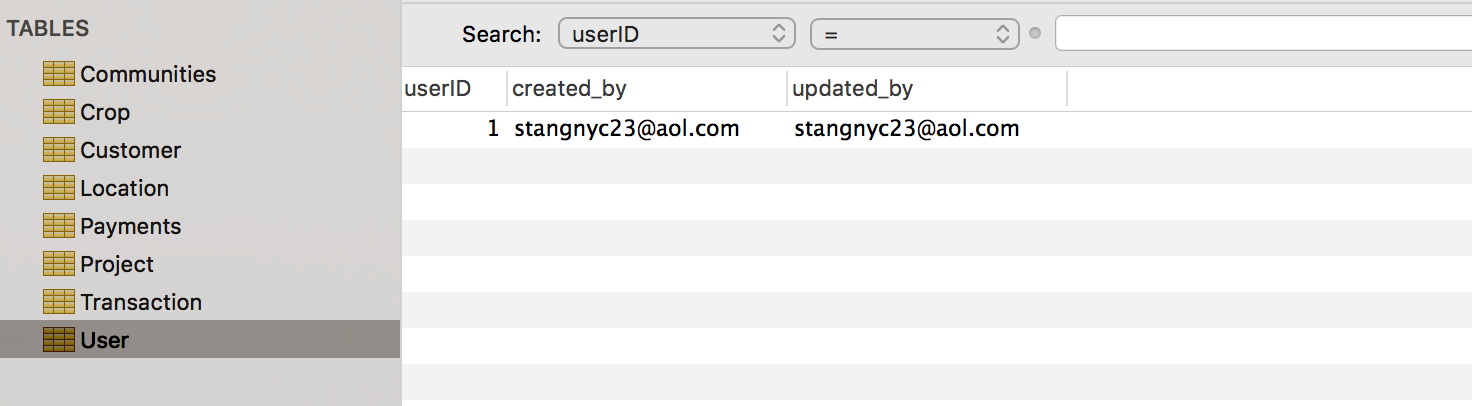
Further, I will import the data into DB using the CSV files. First, I will fill in the child tables and then the parent to avoid any foreign key constraint errors.

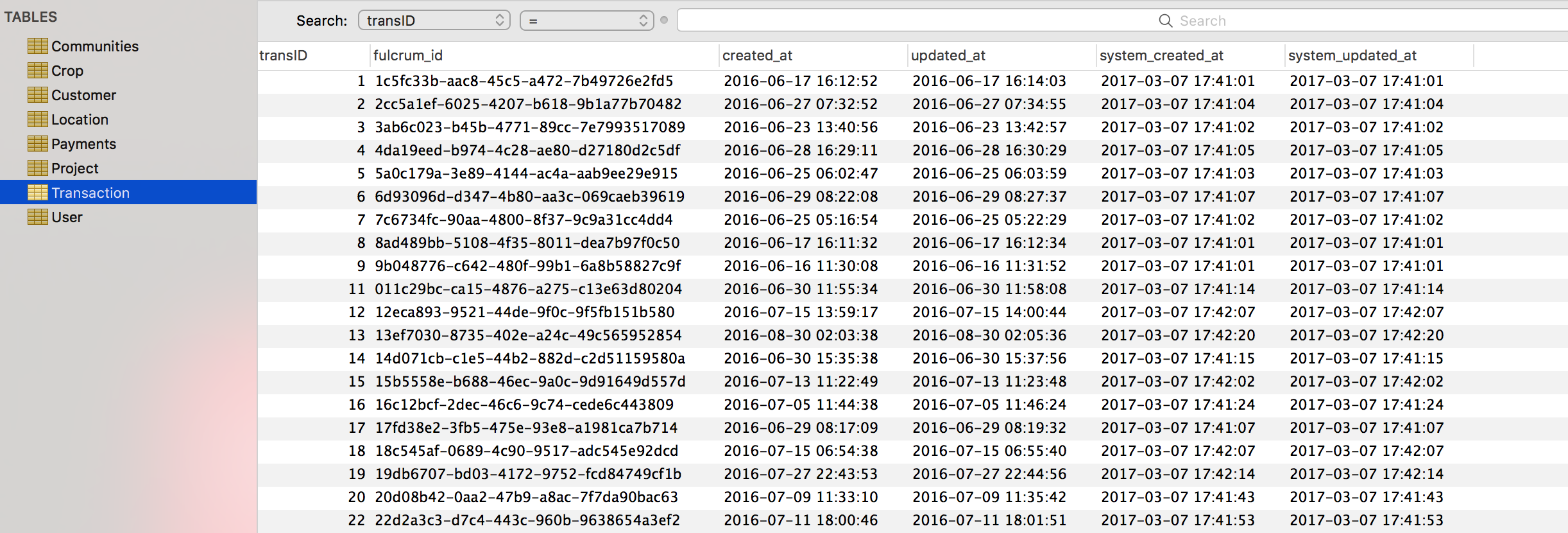


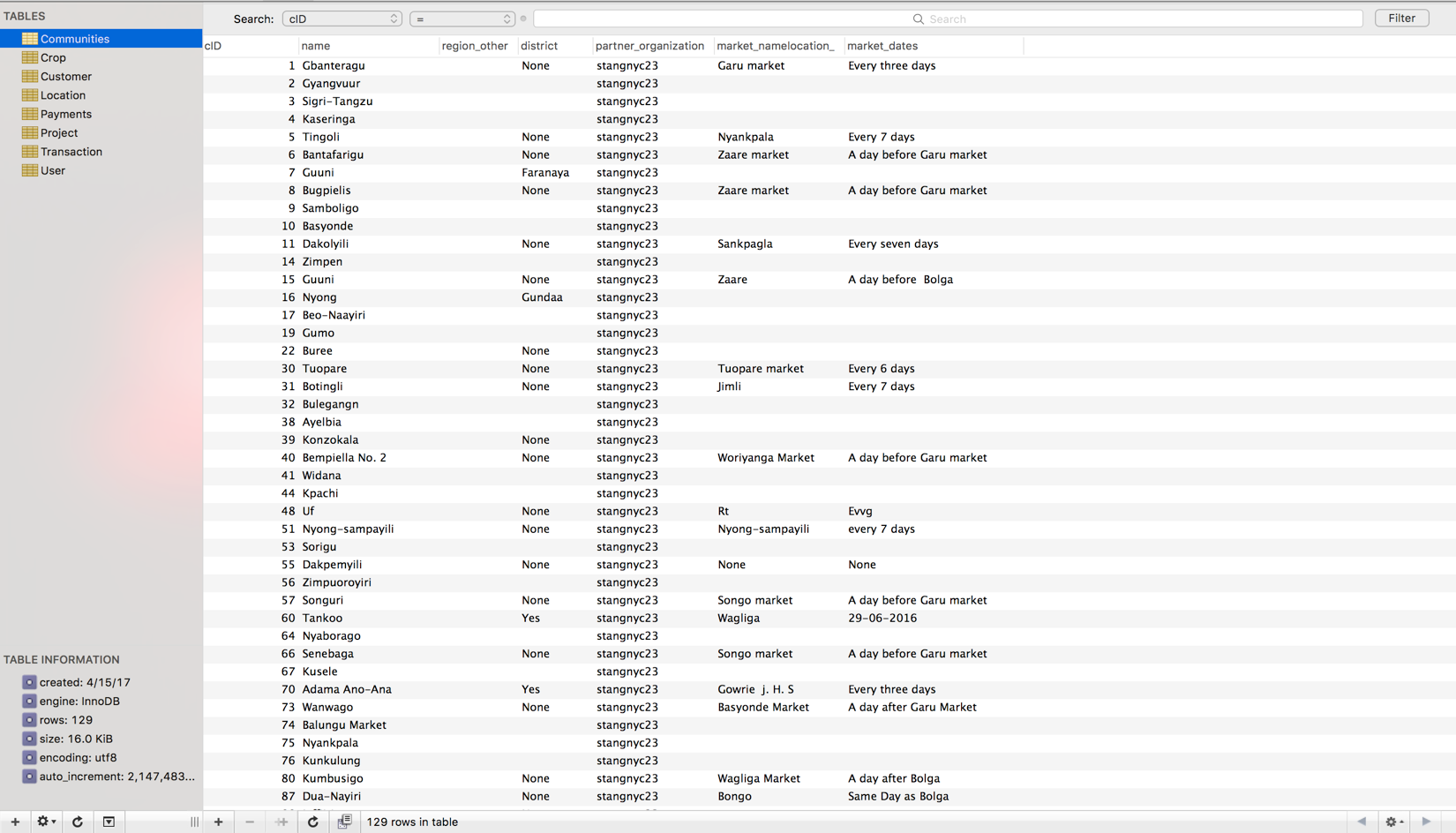












*Note: Since I am using free trial on cloud, it won’t allow me to upload all the data into the database. Thus, I created a local replica and inserted data into it.*

Limitations to this database:

* Since the data from API’s is different. We need to add a middleware layer to map these data transformations.

Advantages:

* It is a Normalized database thus, will is efficient in both speed as well as space complexity.
* Available on cloud. Thus, we get all the benefits of using cloud services to interact with this database.