Raj Deliwala

5/1/2017

Professor David Gerstl

Project Part 3

**Table of Contents**

Gathering raw data………………………………………………………………..2

Unnormalized columns & Raw create……………........................….......…..3 & 4

Normalizing the data & ERD……………………………………………………..5

Create normalized tables & alter tables…………………………………………..6

Data count………………………………………………………………………...7

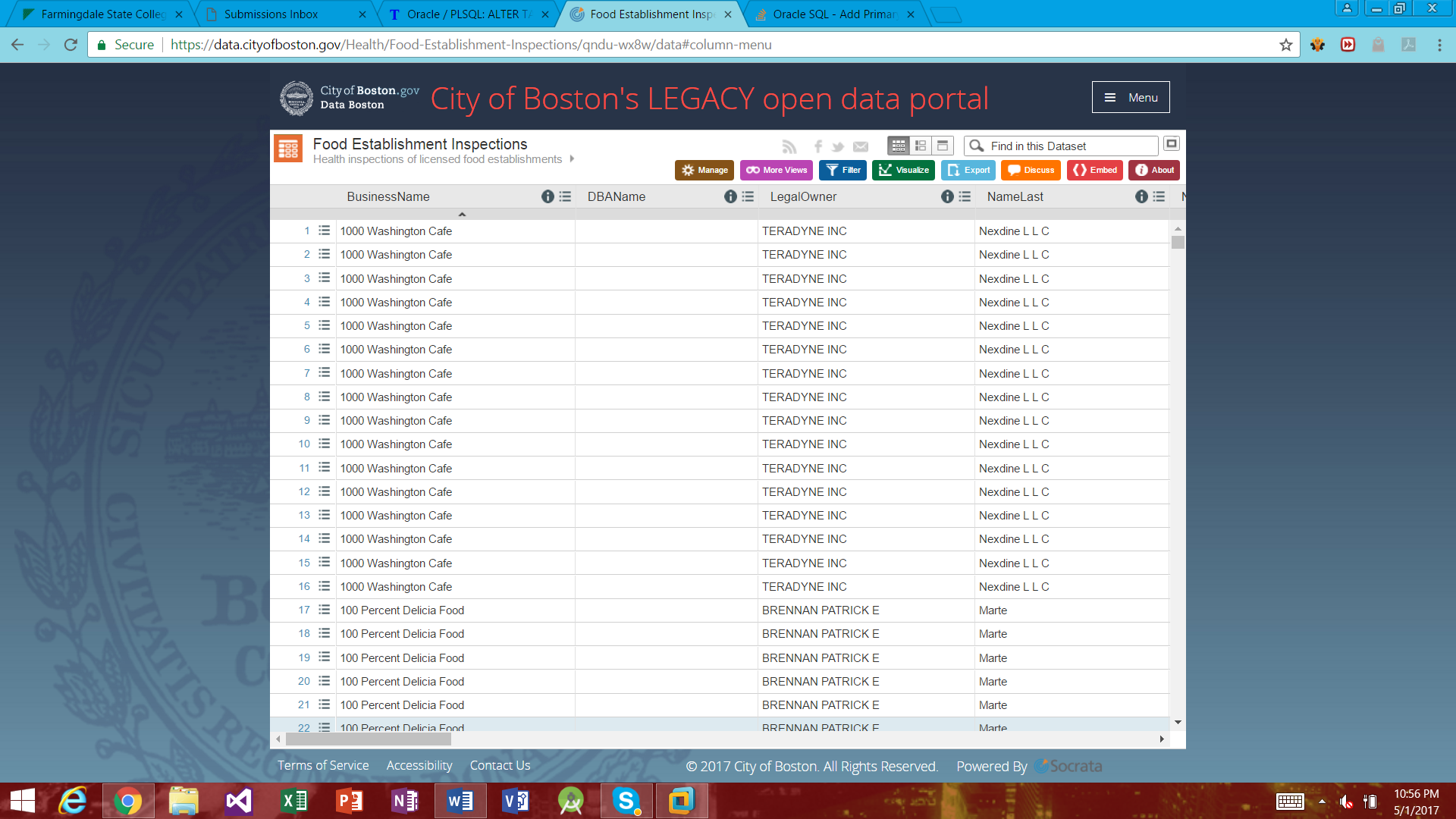
Non-trivial questions……………………………………………………….9 & 10

**BOSTON: Food Establishment Inspections**

This database was created to inspect and regulate all legal licensed food establishments in the state of Massachusetts. This database is constructed by having food inspectors travel to all establishments and perform a proper evaluation. This is proper information to store in a database for legal purposes, there is a certain amount of violations or violation level that if crossed, causes that establishment to be put under punishment.

For the first part of any data set, you should look at it and analysis it checking if it requires normalization. When looking at our data set we can multiple columns or redundant information. Therefore, this is a good data set to play around with.

*References:* <https://data.cityofboston.gov/Health/Food-Establishment-Inspections/qndu-wx8w/data>



The following is the current **Unnormalized data** in the database and what each stands for:

*This is done so we can understand what each column means and it’ll help us normalize the data later on*

**RESTID** – ID created to stand as a CAMIS type column that will be unique, explained below why need in the normalize section

**BusinessName** : This is the column for the establishment’s name

**DBAName** : Extended name, example : BusinessName – Comella’s & DBAName – 1844 Inc

**LegalOwner** : The legal owner of the given establishment

**NameLast** : The last name of the legal owner of the establishment

**NameFirst** : The first name if the legal owner of the establishment

**LICENSENO** : Restaurant license number given to the establishment

**ISSDTTD** : Issued date for given license

**EXPDTTD** : Expire date for given license

**LICSTATUS** : Column stating whether the establishment has an active or disabled license

**LICENSECAT** : License category that corresponds with the license number of an establishment

**DESCRIPT** : Small description corresponding with the given license of an establishment

**RESULT** : Result of that inspection

**RESULTDTTM** : Date that the inspection result was given

**Violation** : Code for the violation that occurred

**ViolLevel** : Level of violation that was found

**ViolDesc** : Brief description on the given violation

**VIOLDTTM** : Date the violation was found

**ViolStatus** : Given status of the violation, ether Pass or Fail

**StatusDate** : Date of the given status

**Comments** : Field used to enter any extra comments on the inspection

**Address** : Given address of the establishment

**City** : Given city of the establishment

**State**: Given state of the establish, all should just be MA since this is an Inpsection database from there

**Zip** : zip code of the given establishment

**Property\_ID** : Property ID that is attached with the establishment

**Location**: Long and Lat coordinates that pin-point the establishment’s location

**Original create table:**

CREATE TABLE bostonInspec (

BUSINESSNAME VARCHAR2(200),

DBANAME VARCHAR2(60),

LEGALOWNER VARCHAR2(100),

NAMELAST VARCHAR2(75),

NAMEFIRST VARCHAR2(75),

LICENSENO NUMBER,

ISSDTTM VARCHAR2(30),

EXPDTTM VARCHAR2(50),

LICSTATUS VARCHAR2(20),

LICENSECAT VARCHAR2(50),

DESCRIPTION VARCHAR2(200),

RESULT VARCHAR2(20),

RESULTDTTM VARCHAR2(50),

VIOLATION VARCHAR2(50),

VIOLATIONLEVEL VARCHAR2(25),

VIOLATIONDESCRIPTION VARCHAR2(200),

VIOLATIONDTTM VARCHAR2(50),

VIOLATIONSTATUS VARCHAR2(50),

STATUSDATE VARCHAR2(50),

COMMENTS VARCHAR2(3000),

ADDRESS VARCHAR2(80),

CITY VARCHAR2(40),

STATE VARCHAR2(50),

ZIP NUMBER,

PROPERTYID VARCHAR2(50),

LOCATION VARCHAR2(100)

);

**What we need to do to normalize the data:**

To normalize the data in this set, we’re going to break it down into 4 different tables. Similar to the NYS Restaurant Inspection data base we did in class, I plan to break this database down in a similar fashion. To normalize the data, I’m going to break it down into 4 normalized tables: Establishment, License, Violation, and Inspection\_Line\_Item.

**Note:** *Since there is no unqiue ID like CAMIS in the NYC dataset, were going to create a column called RESTID that will hold the rownum and create a unqiue key like CAMIS.*

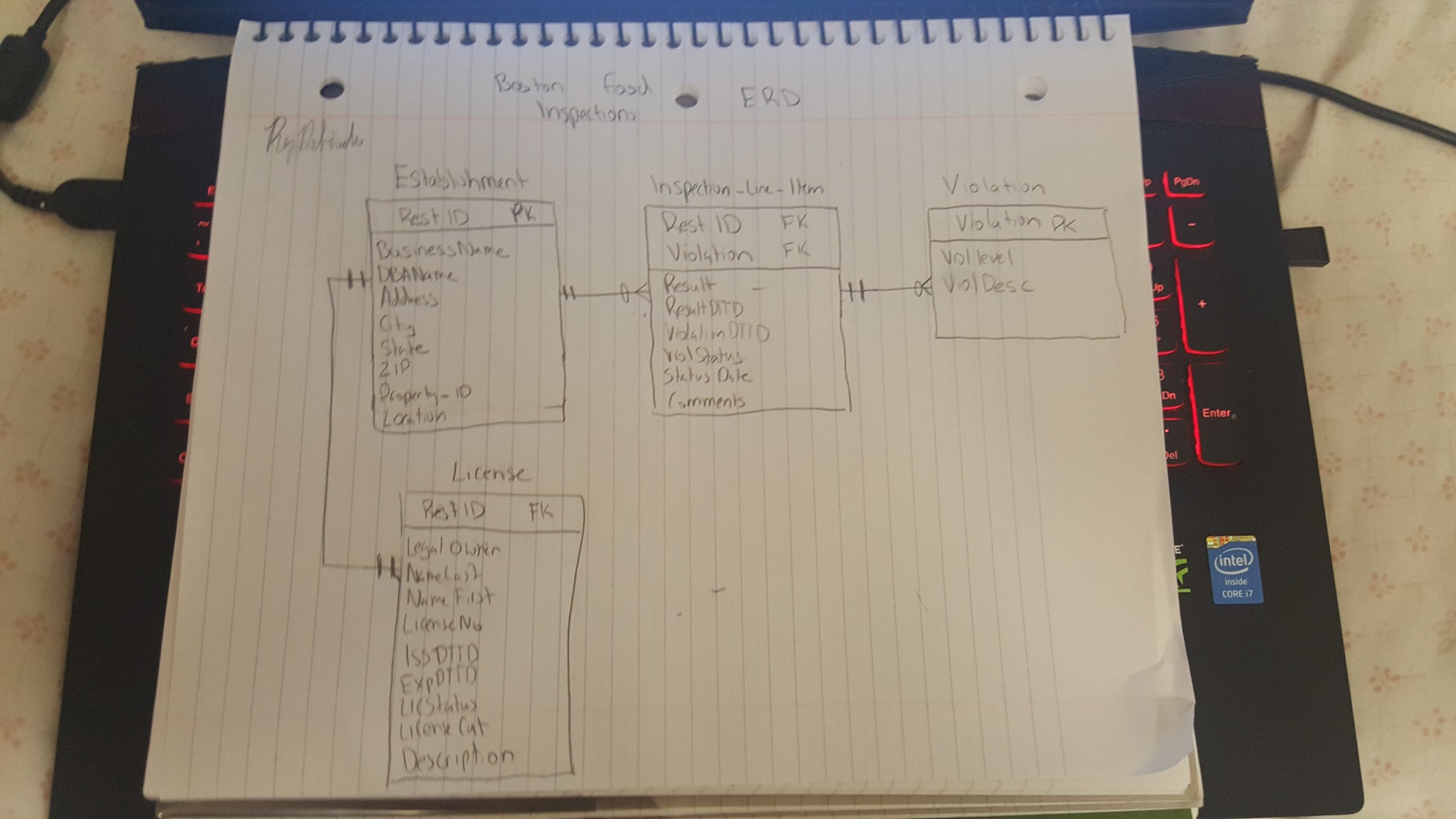
**Establishment – (RESTID**, **BusinessName, DBAName, Address, City, State, Zip, Property\_ID, Location)**

**License – (RESTID, LegalOwner, NameLast, NameFirst, LICENSENO, ISSDTTD, EXPDTTD, LICSTATUS, LICENSECAT, DESCRIPT)**

**Violation – ( Violation, ViolLevel, ViolDesc)**

**Inspection\_Line\_Item– ( RESTID, RESULT, RESULTDTTM, Violation, VIOLDTTM, ViolStatus, StatusDate, Comments)**

**ERD:** *Created to show how to normalize the raw data and what are the relationships between them*

****

**CREATE TABLE STATEMENTS TO CREATE NORMAILIZED TABLES:**

*New create tables for the normalized data*

create table ESTABLISHMENT as (select restid, businessname, dbaname, address, city, state, zip, propertyid, location from bostonInspec);

create table LICENSE as (select restid, legalowner,namelast,namefirst,licenseno,issdttm,expdttm,licstatus,licensecat,description from bostonInspec);

create table VIOLATION as (select violation, violationlevel, violationdescription from bostonInspec);

create table INSPECTION\_LINE\_ITEM as (select restid, violation, result, resultdttm, violationstatus,statusdate, violationdttm, comments from bostonInspec);

**Alter tables to add to the tables:**

*Alter table*

alter table establishment add constraint establishment\_pk primary key(restid);

alter table license add constraint license\_fk foreign key(restid) references establishment(restid);

alter table violation add constraint violation\_pk primary key (violation);

alter table inspection\_line\_item add constraint inspection\_pk primary key (restid,violation),

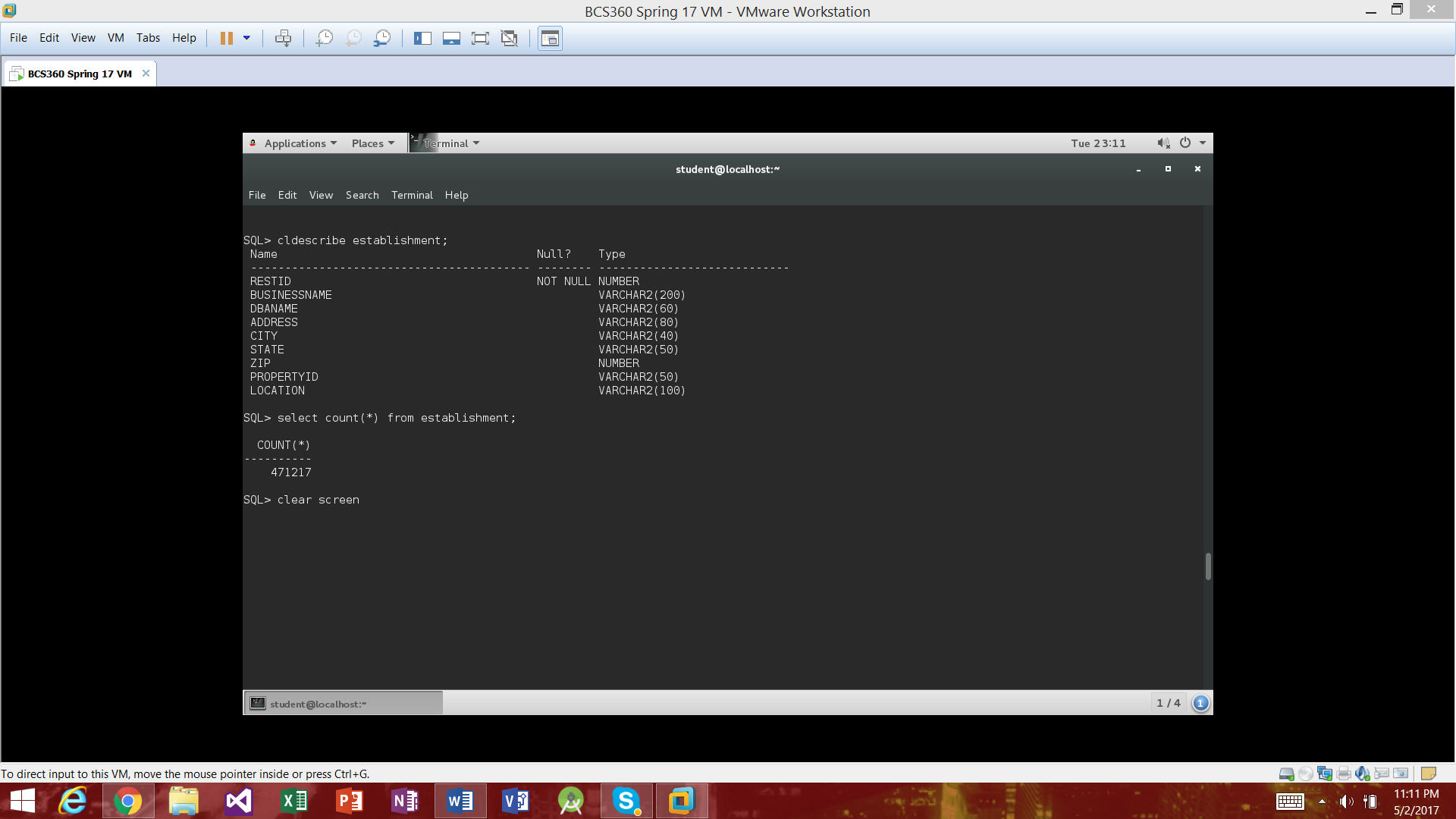
add constraint inspection\_establishment\_fk foreign key (restid) references establishment (restid),

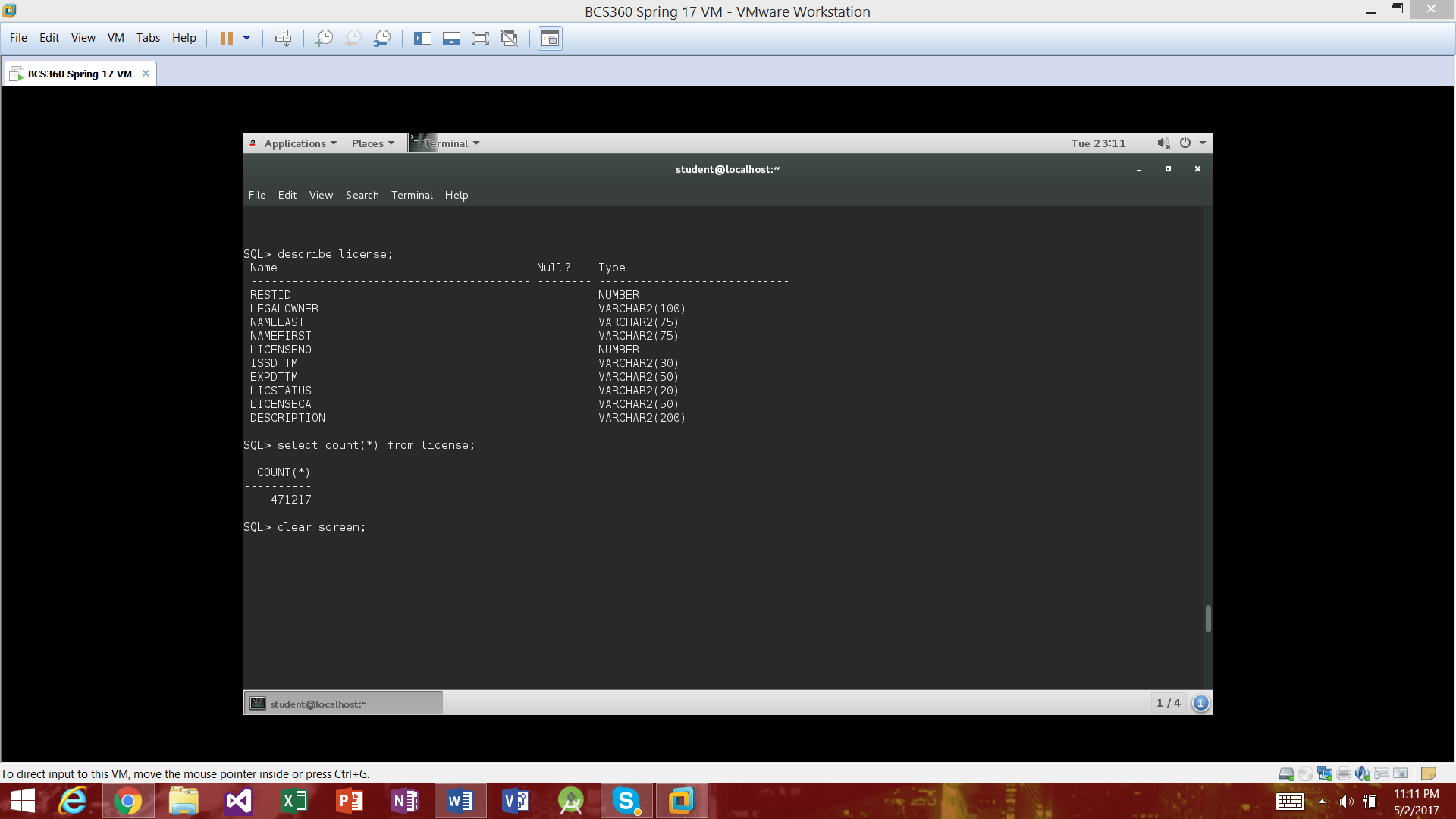
add constraint inspection\_violation\_fk foreign key (violation) references violation (violation);

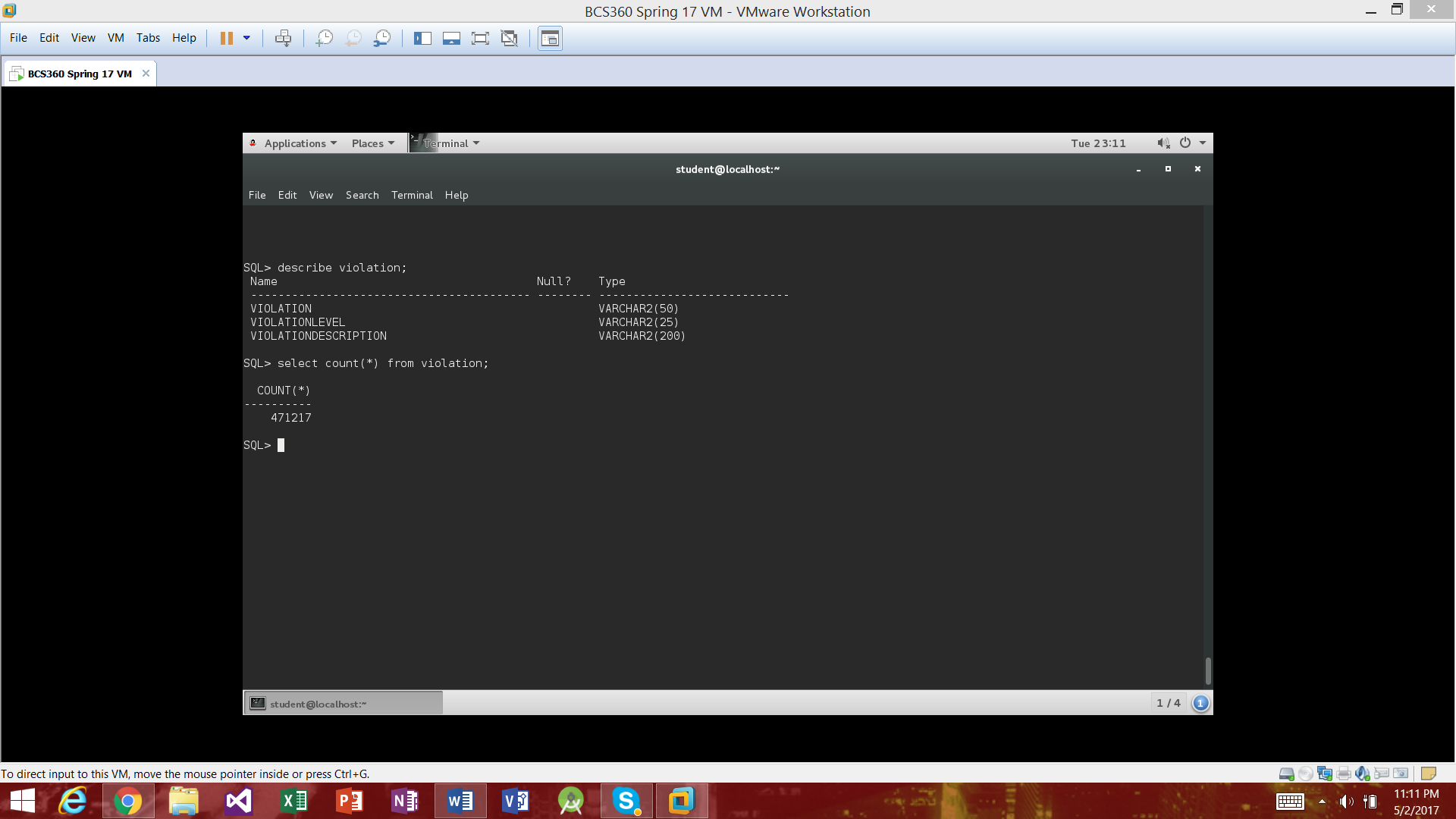
**Data count and screenshots:**

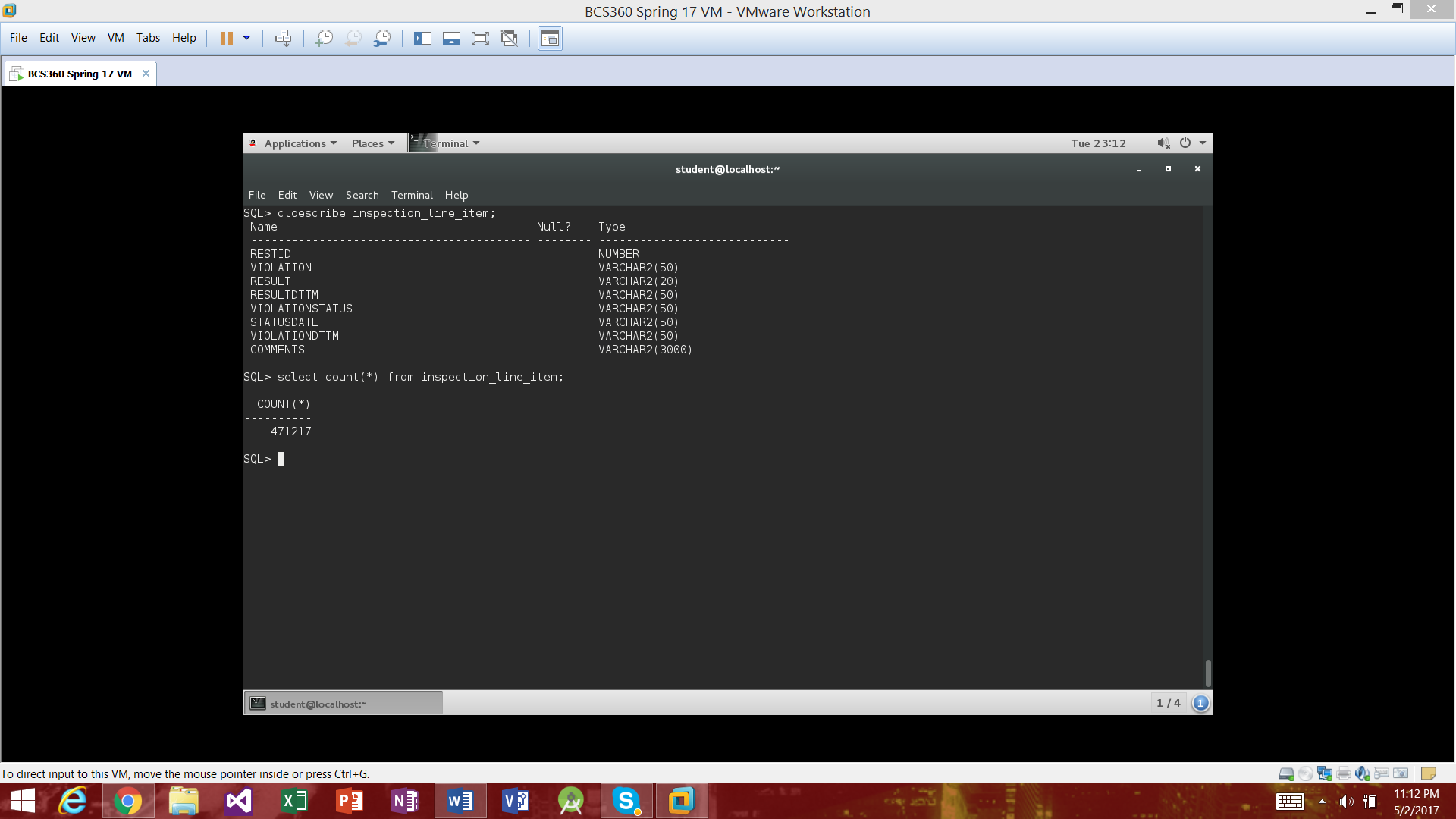
In my case the count did not vary from unnormalized data to normalized data, because the keys were not able to work properly under this data set due to the large amount of null data.

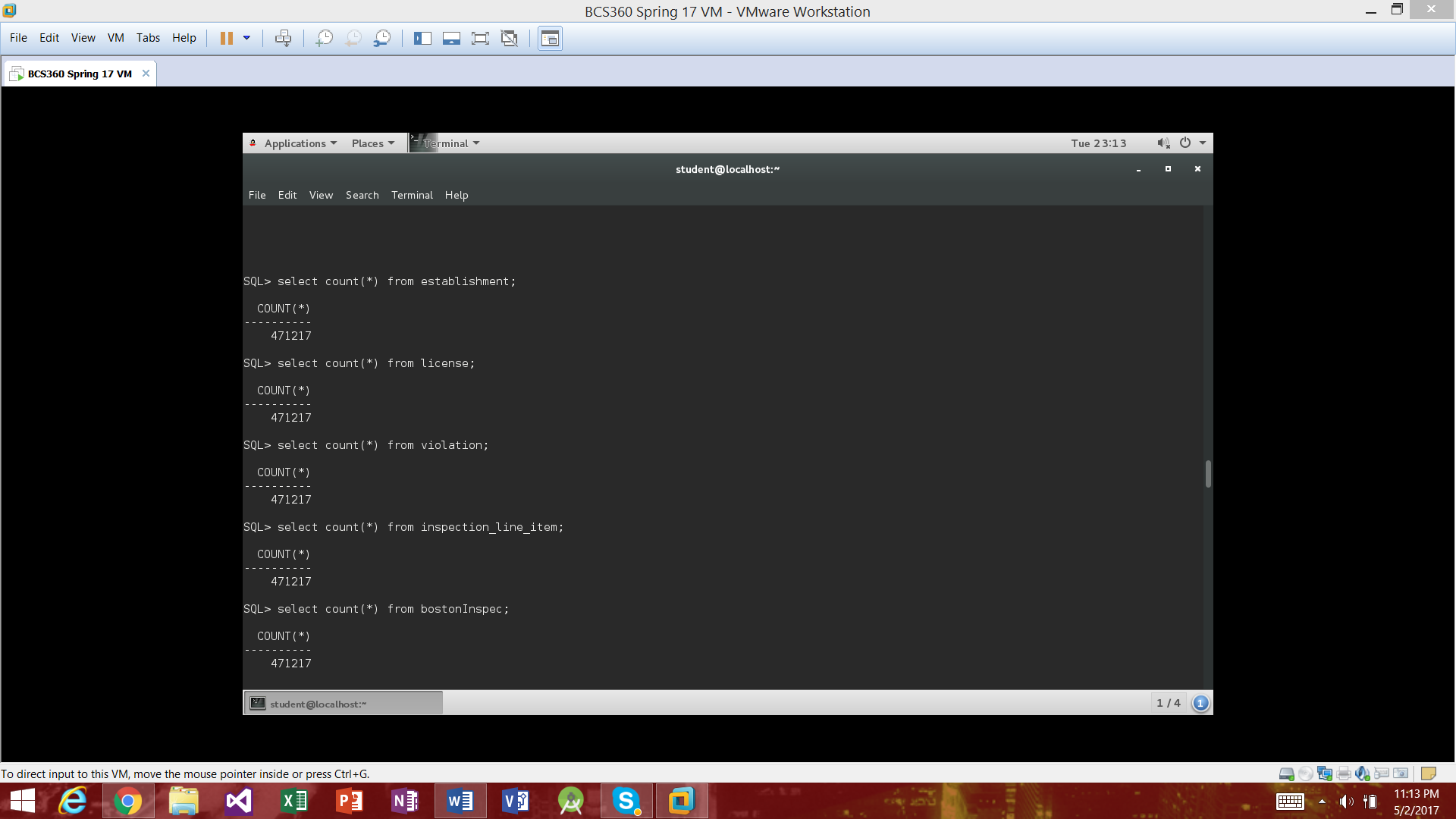
In a case where the keys could work, you would see a less count due to the user being able to get rid of the null data.











**Non-Trivial questions**

Now that we have the data split up and normalized we can ask it some non-trivial questions and learn few things about Boston food establishments

1. Let’s find all the Business names and owners with a count of their active licenses.

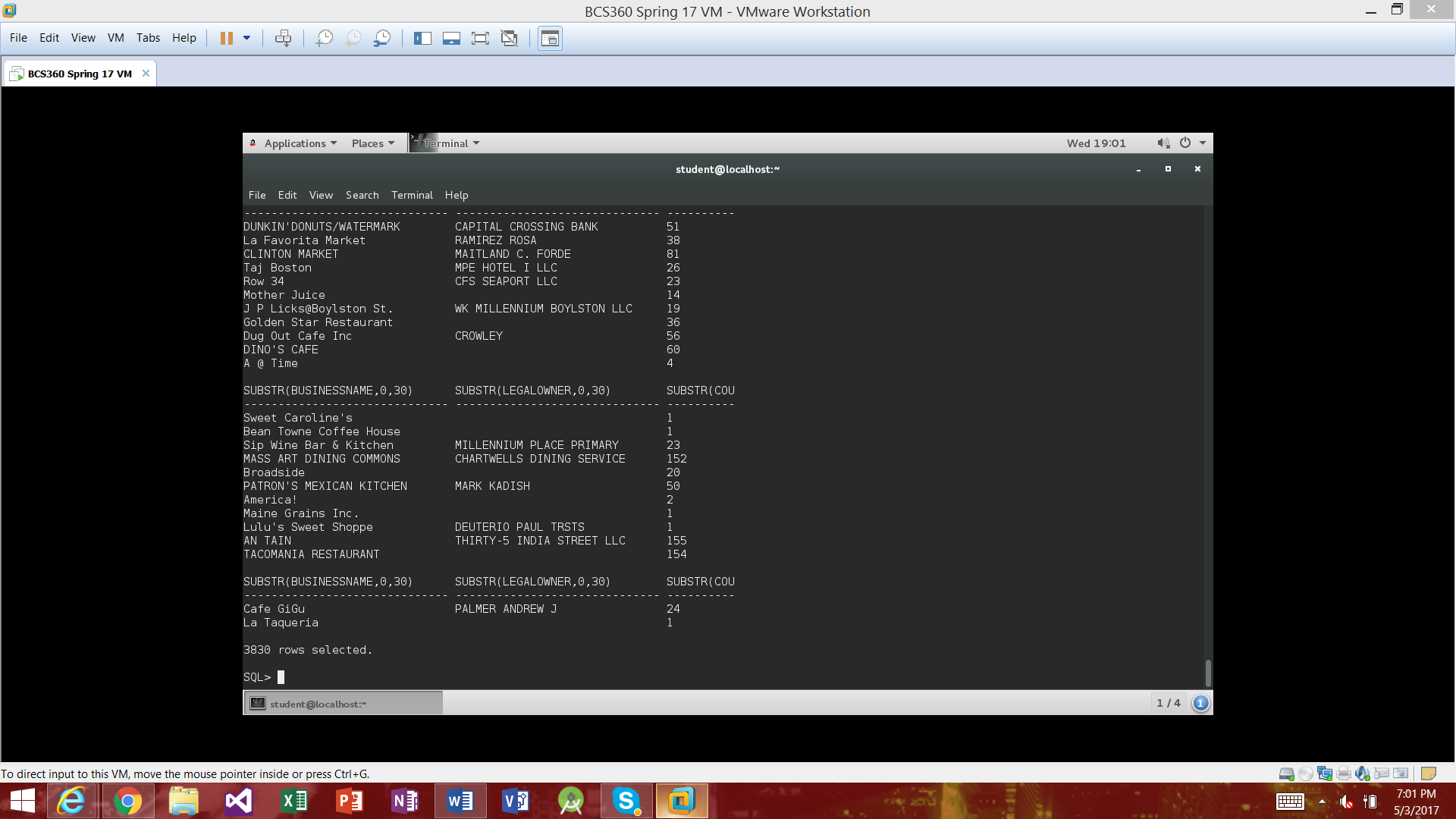
Select subtr(businessname,0,30), substr(legalowner,0,30), substr(count(licstatus),0,10)

From establishment e join license l on

e.restid = l.restid

where licstatus = ‘Active’

group by businessname, legalowner;



1. Let’s find all the businesses that have a passing violation status and their violation level

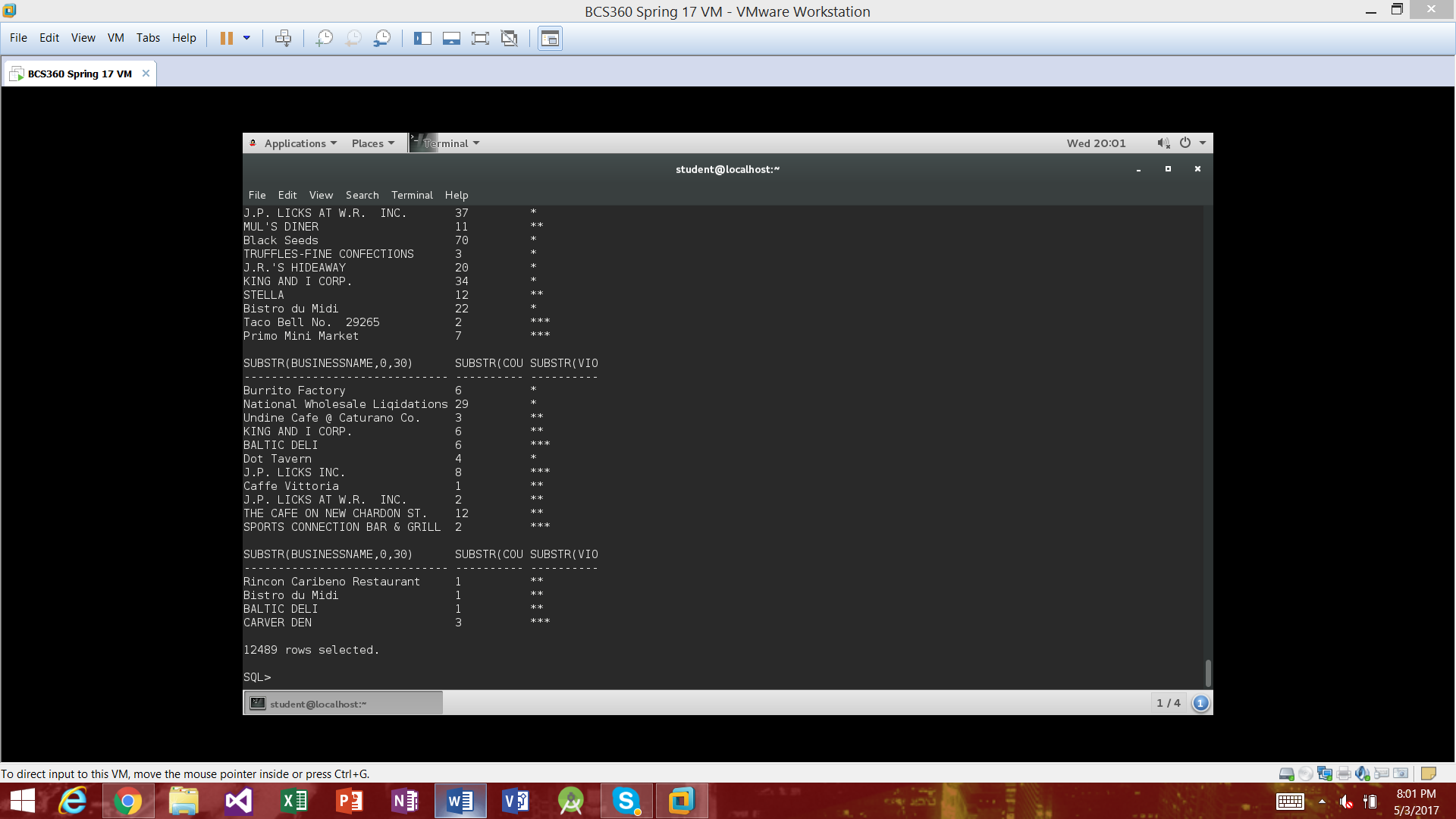
Select substr(Businessname,0,30), substr(count(violationstatus),0,10), substr(violationlevel,0,10) from establishment e join Inspection\_line\_Item L on

e.restid = l.restid

join violation v on

l.violation = v.violation

where violationstatus = ‘Pass’ group by businessname, violationlevel;



*Checklist is the table of content at the beginning of this paper*