Practicum – 1

Contact Tracing System for Epidemiologists and Epidemiological Research

Aman Batra

NUID: 001877232
Class: CS5200
Academic Term: Summer-2, 2020
Instructor: Dr. Martin Schedlbauer
Email : Batra.am@northeastern.edu
No Group – Individual Submission

1. Problem Definition – Contact Tracing

1.1 What is Contact Tracing?

According to WHO: Contact tracing is the process of identifying, assessing, and managing people who have been exposed to a disease to prevent onward transmission.

1.2 What is the purpose of Contact Tracing?

When systematically applied, contact tracing will break the chains of transmission of COVID-19 to prevent future waves or surges of cases, and to enable us to get back to work in a much safer way. Contact Tracing is an essential public health tool for controlling the virus.

1.3 How does Contact Tracing work?

Contact tracing for COVID-19 requires identifying people who may have been exposed to COVID-19 and following them up daily for 14 days from the last point of exposure. The goal is to create a spider web of coronavirus transmission

Multiple Approaches To Trace Contacts:

1. An application approach, Continuous subject monitoring and data gathering is achieved using a mobile-application. Patients use the application to self-assess symptoms and report their interactions with other contacts, which can then be notified via the app, and put in incubation. For Example, a portable contact tracing application with real-time threat notifications based on GPS location/Bluetooth tracking, daily self-assessments, and contact reporting.

2. A general approach, where infected person contacts local Public Health Authorities and notifies them, PHAs then takes report of the person via calls & interviews, and prescribes test or medical assessments. Daily reports and health updates are taken by the assigned personnel until the incubation period is active. All the contract tracing is done manually by the PHAs, who then feed the data into a central reporting system.

3. A combined approach, where both self-assessments and manual assessments are possible. Patients choose if they would like to contact and schedule meetings with PHAs or prefer the use of a real-time contact tracing app regularly. PHAs gather and anonymise all the data, and put it in a graph database. The graph databases are then analysed to create heatmaps of the COVID-19 affected areas, which are later turned into containment zones.

2. Most Efficient Approach - Contact Tracing (Combined Approach)

Combined approach assumes that the local, regional, or national Public Health body of a geographical area, has offered three ways, to keep a check on COVID-19 epidemic: a mobile application, a website, and a dedicated COVID-19 helpline to implement contact tracing measures.

1. **Automated**: A self-assessment and self-reporting based portable application, capable of running on multiple types of devices such as mobile phones, tablets, and laptops with real-time geolocation and Bluetooth based proximity tracing of other users, running the same application of their devices. Each user's ***HealthStatus*** tokens are broadcasted within a 30 metres radius, and every user of the application in the proximity range will receive these tokens. If a sick user is nearby, all the users in the vicinity will get a threat alert. The total interaction time or visit time of a user will be recorded whenever he or she, meets or passes by, a sick person, or visits a place. If a user is not feeling and suspects that he or she might have been exposed to COVID-19, then there is a self-assessment option which contains a predefined set of questions, that can predict the likelihood of infection. If the assessment score bypasses the defined threshold, all users in the vicinity are notified of potential threat; Local health authority(LHA) is notified via the application. The LHA then prescribes a suitable COVID-19, and if the user tests positive, an incubation period of 14 days is initiated, which contains daily self-assessments, self-quarantining, and self-reporting of every place visited, every person in touch, and every notable interaction made within the past 14 days.

2. **Semi-Automated:** Some users have privacy issues when it comes to using applications that continuously record user data, and keep surveillance over their activities. For such users, a website or a similar app without monitoring is a better option, to implement contact tracing. When such a user feels sick, he can go to the LHA website, and take a self-assessment test, if the test results indicate potential infection, their identity is anonymised and location data is fed into the central database(common to all strategies). The user has to take a daily self-assessment until he or she is marked 'healthy' again, and all the contact reporting is done on the website. The central database is used to create heatmaps of coronavirus stricken areas, the website users can manually check the heatmap zones online, whereas it is inbuilt in the application.

3. **Manual:** For users, who have no viable means to use the website or application, there is a dedicated helpline number, which takes care of daily assessments, and contact reporting. Things are done manually, via phone conversations or administered meetings, and all the gathered data is then manually uploaded to the database.

3. Database Tools Specification

Overview

The database for a Contact Tracing system is can be implemented using many relational and non-relational DBMS such MySQL, PostgreSQL, MongoDB, Neo4j, Oracle DB. Parts of the project can be implemented using graph-databases, because they will be best suited for running depth queries and discovering links at greater depths, but for this particular Practicum, we will be using MySQL as a DBMS tool ubiquitously, and R Studio for running Analytics. MySQL has many advantages and some limitations, as given below:

Advantages of MySQL:

1. Open source, inexpensive and readily available.
2. Industry Standard, and very popular.
3. Extensive support available online.
4. Ease, Intuitiveness and Usability
5. Outstanding InnoDB engine.

Drawbacks:

1. Scalability issues can arise with time.
2. Not very easy to debug.
3. Does not support very large databases efficiently.

Advantages of R:

1. Open source, Platform Independent
2. Rapid, and quality plotting
3. Non-Coder friendly, anyone can start plotting within a hours
4. Rich and continuously growing sets of packages (>10000) in the CRAN repository

Drawbacks:

1. R Utilizes more memory as objects are stored in the memory.
2. Slower than other programming languages like Python and Matlab
3. Does not support very large scale applications efficiently.

4. Overall Assumptions & Constraints.

Assumptions:

1. Only some parts of the whole application will be reflected in the relational database, some data will stored locally on the host devices, such as state variables, local variables, events data, device permissions details, etc.
2. There can be many use cases, but the database is designed keeping in mind only some of these use cases, hence some parts of the database can be missing For Example. The assessment-survey module can have 4 more classes, but we are only using one for now.
3. The depth of queries will be set to 5, because MySQL is not a graph-database, it takes a good amount of time and processing power to create Joins, and make connections.
4. Application logic and host application will be created at a later time.

Constraints:

- **Domain Level:**
    1. **Varchar** for string data
    2. **Integer** for whole numbers
    3. **Boolean** for binary choices (True or False, Yes or No, Correct or Incorrect, these kind of choices will be implemented using 0s & 1s
    4. **Enum** for Lists or categorical attributes
    5. **Text** for descriptions

- **Referential Integrity:**
    1. *Place* extends to *Interactions, AppUser, Visits, PublicHealthAuthority*
    2. *Person* extends to *AppUser* and *PublicHealthWorker*
    3. *AppUser* extends to *Place, Interactions, Assessment, HealthReportCheck* and *Person*
    4. *UserEvents* extends to *Visits*, *Interactions* and *ContactHistoryLog*
    5. *PersonNotification, AppUserEmail, PersonPhone* are linking tables.

- **Entity Integrity**
    1. All classes have primary keys, all primary keys are set to NOT NULL
    2. Key constraints are enabled SET FOREIGN_KEY_CHECKS = 1

5. UML Class Diagram

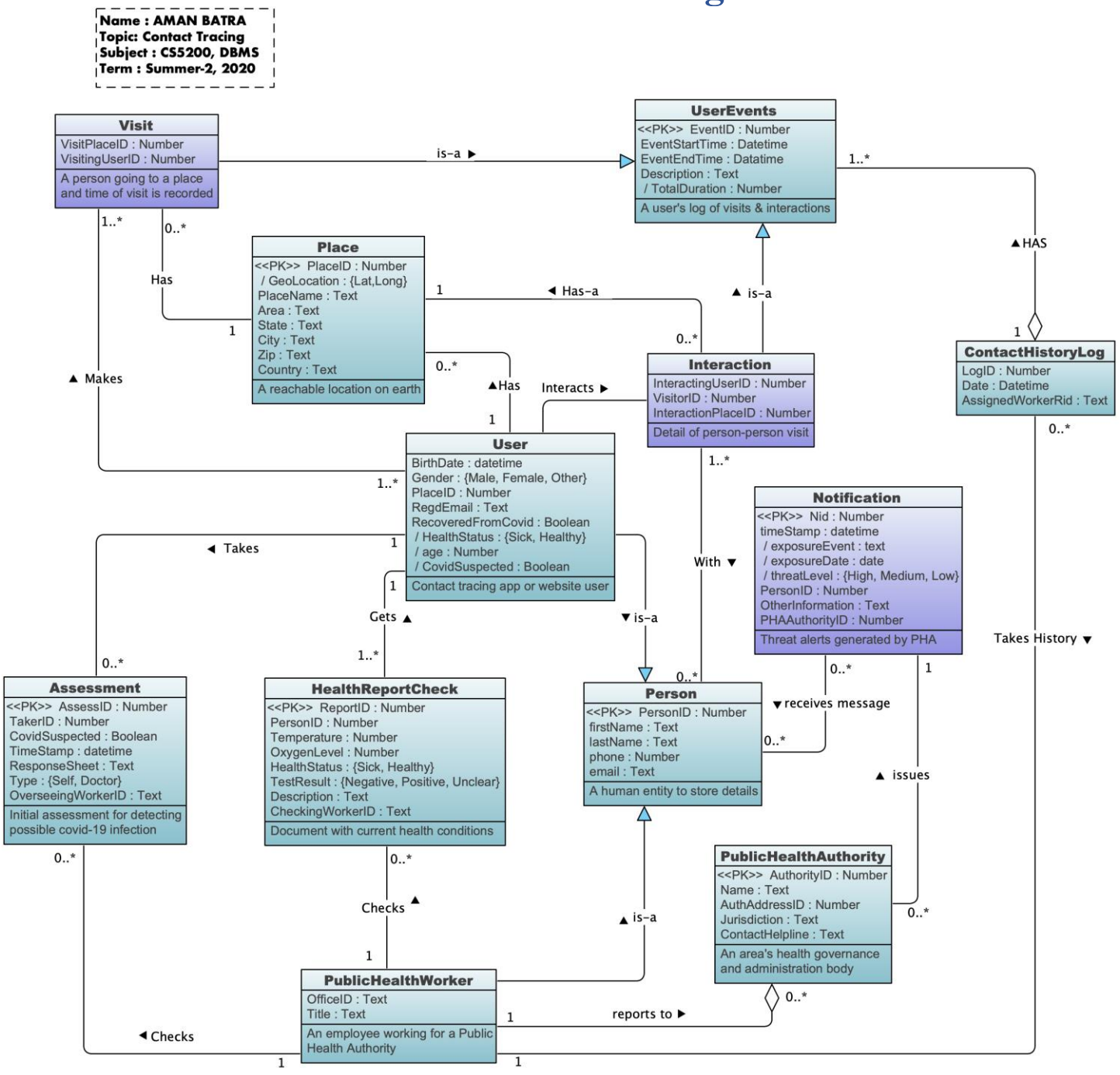


Diagram 1 - Conceptual UML Class Diagram, for a COVID-19 Contact Tracing System to aid epidemiological research.

Design Tool Used: Visual Paradigm Community Edition for Mac
Link to Files:

6. Entity Relationship Diagram

Link to LucidChart: https://app.lucidchart.com/documents/view/982d3b7d-538a-4993-9511-126f0853bb6e/0_0#

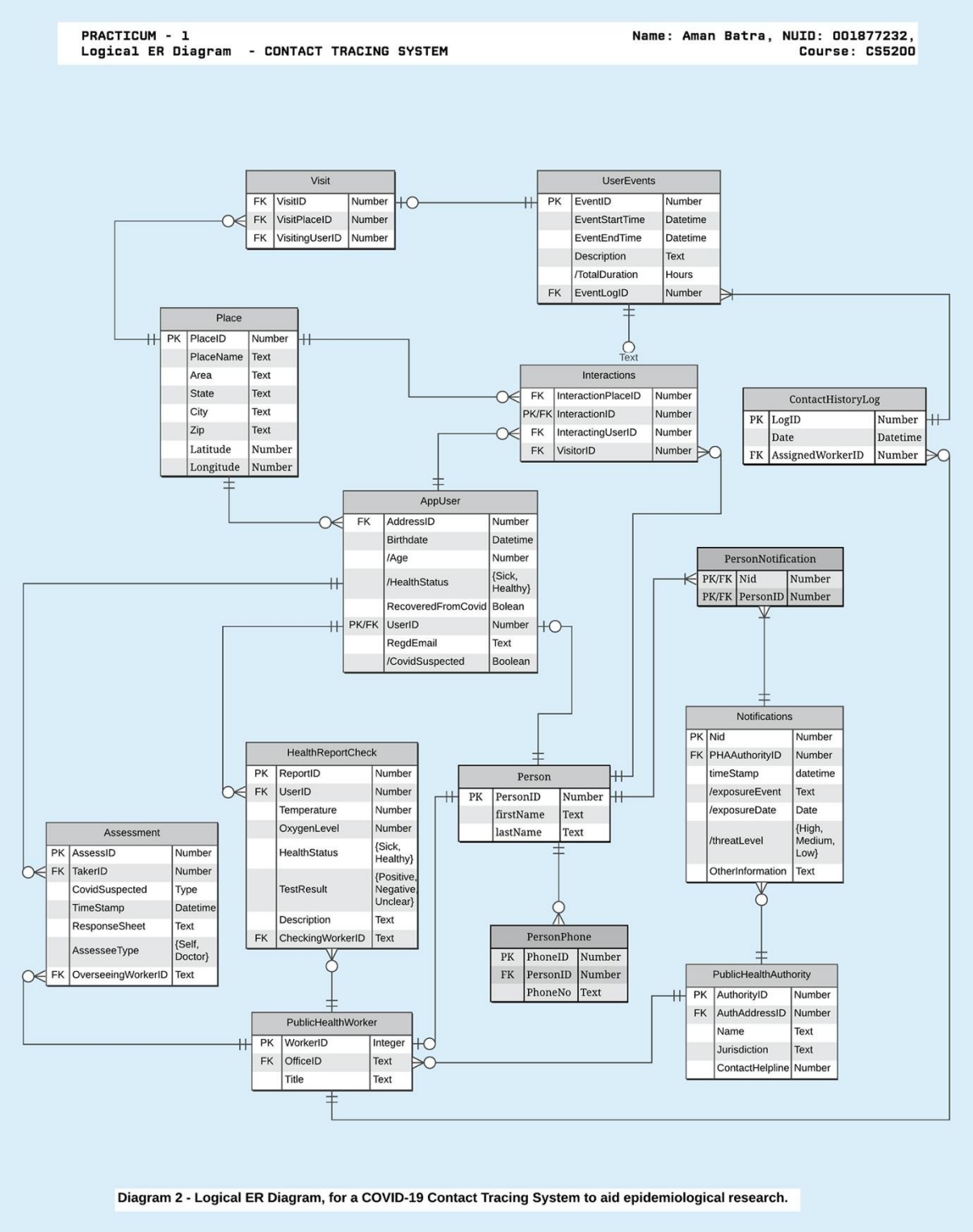


Diagram 2 - Logical ER Diagram, for a COVID-19 Contact Tracing System to aid epidemiological research.

7. Schema Generation & Normalization Check Results

Relational Schema:

Place(*placeint*,placename,streetarea,city,state,zip,latitude,longitude)

Person(*PersonID*,firstName,lastName)

PersonPhone(PhoneNo,PersonID)

AppUser(*UserID*,*AddressID*,RecoveredFromCovid)

AppUserEmail(EmailID, UserID)

PublicHealthWorker(WorkerID,title,*officeID*,PublicAuthID)

PublicHealthAuthority(*AuthorityID*,AuthAddressID,Name,Jurisdiction)

PublicHealthAuthorityHelpline(HelplineNo,AuthorityID)

UserEvent(*EventID*,EventStartTime,EventEndTime,Description)

Visits(VisitID,VisitingUserID,VisitPlaceID)

Interactions(InteractionID,interactingUserID,VisitorID,InteractionPlaceID)

Notifications(*Nid*,PHAAuthorityID,timestamp,OtherInformation)

HealthReportCheck(*ReportID*,UserID,Temperature,OxygenLevel,HealthStatus,TestResult,Descriptions,CheckingWorkerID, ReportDate)

Assessment(AssessID,TakerID,CovidSuspected,TimeStamp,ResponseSheet, AssesseeType, OverseeingWorkerID)

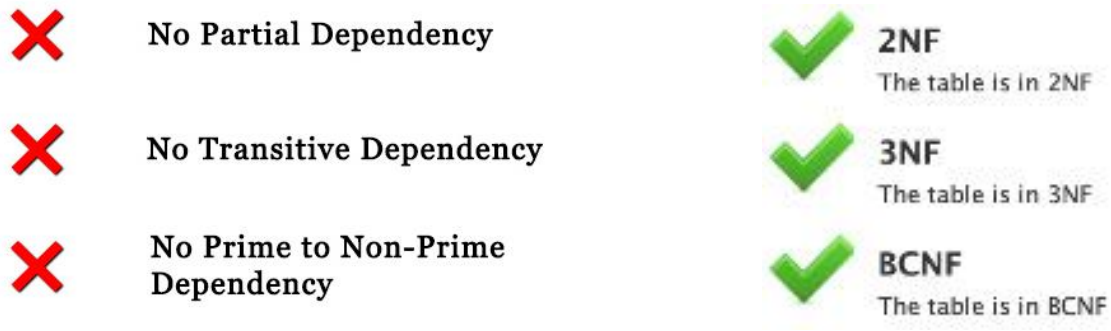
PersonNotification(Nid, PersonID)

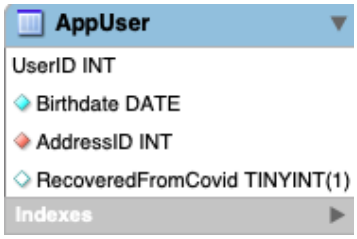
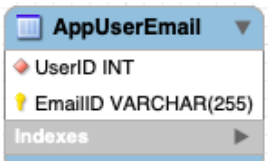
Normalization to BCNF

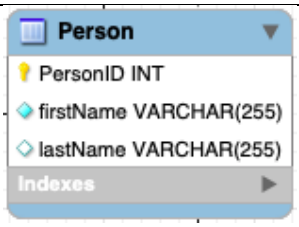
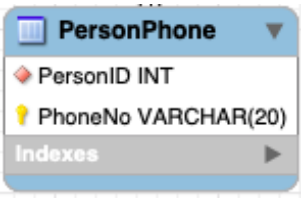
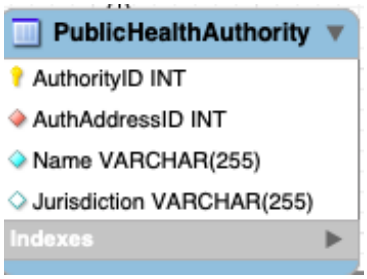
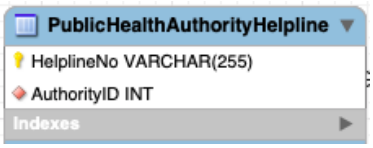
This following table lists out every relation in the database and provides proof to make sure its in BCNF. There Is no need to prove lower normal forms like 1NF, 2NF and 3NF because, if a relationship in BCNF, it IMPLIES that it is already normalized in lower forms.

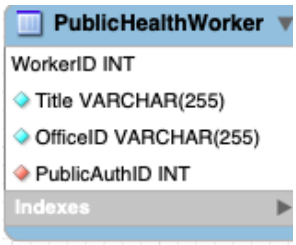
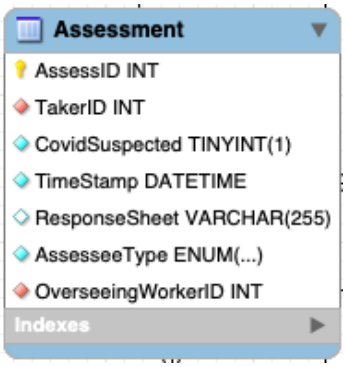
The relationships shows below, comply with all of the following criterion, needed for validating BCNF.

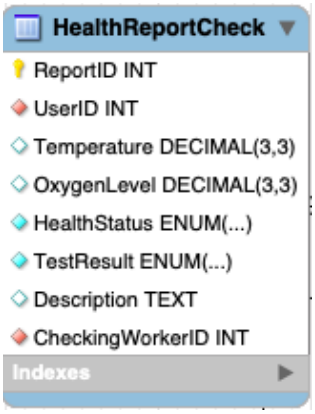
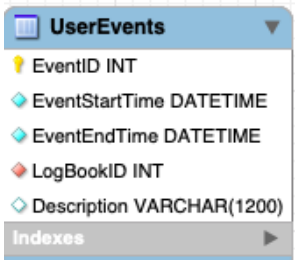
1. Every relationship has a valid candidate key as their determinants(All determinants are candidate keys)
2. There is no partial dependency of any kind
3. No composite candidate keys with overlapping attributes
4. No multivalued attributes exist
5. No transitive dependency.

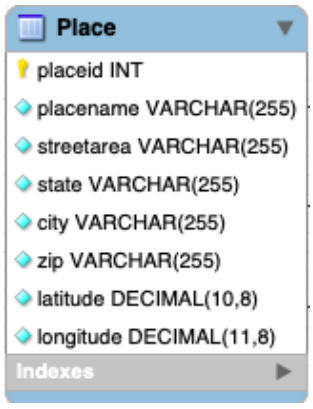
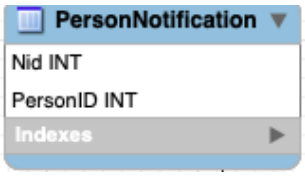
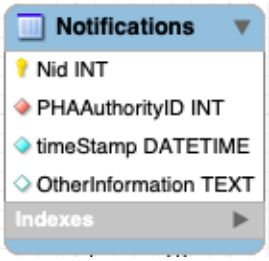
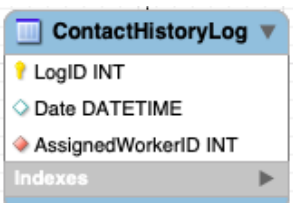


S.No.	Table Name	All Functional Dependencies
1	 UserID=>addressid UserID=>=>name UserID=>birthdate	personid → addressid addressid → personid name birthdate • **UserID** is a candidate key and Primary key which uniquely identifies Birthdate, AddressID, RecoveredFromCovid. • **AddressID** is a Candidate Key, because it can uniquely identify an AppUser • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
2	 EmailID -> UserId	EmailID -> UserId • **UserID** is a candidate key and Primary key which uniquely identifies Birthdate, AddressID, RecoveredFromCovid. • **AddressID** is a Candidate Key, because it can uniquely identify an AppUser • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF

3	 PersonID=>firstname PersonID=>lastname	personid → firstname lastname • **PersonID** is a candidate key and Primary key which uniquely identifies firstname, lastname • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
4	 PhoneNo=>PhoneID	personid → phoneno • **PhoneNo** is a candidate key and Primary key which uniquely identifies PersonID • PersonID is not unique here because a user can have multiple phone numbers. • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
5	 AuthorityID=>AuthAddressID AuthorityID=>Name AuthorityID=>Jurisdiction	authorityid → authorityaddressid name jurisdiction name jurisdiction → authorityid • **authorityid** is a candidate key and Primary key which uniquely identifies all other attributes • name, jurisdiction for a secondary relationship that can identify authorityid • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
6	 helplineno=>authoritid	helplineno → authorityid • **helplineno** is a candidate key and Primary key which uniquely identifies AuthorityID, because a number cannot be shared by multiple authorities in our assumption • authorityID has multiple numbers hence not unique to relationship • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF

7	 Workerid => officeid Workerid => Title Workerid => publicauthid	 • **WorkerID** is a candidate key and Primary key which uniquely identifies Title, OfficeID, publicauthID • **OfficeID** is a candidate key which uniquely identified the PK • PubliAuthID has multiple workers so not unique to relationship • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
8	 AssessId=>takerid AssessId=>covidsuspected AssessId=>timestamp AssessId=>responsesheet AssessId=>assesseetype AssessId=>overseeingworkerid	 • **WorkerID** is a candidate key and Primary key which uniquely identifies all the attributes • Same TakerID is on multiple assessments so not unique to relationship • Timestamp, takerid and overseeingworkerid form a secondary relationship (non-primes to key) • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF

9	 ReportID=>userid ReportID=>temperature ReportID=>oxygenlevel ReportID=>healthstatus ReportID=>testresult ReportID=>description ReportID=>checkingworkerid ReportID=>reportdate	reportid → userid temperature oxygenlevel healthstatus testresult description checkingworkerid reportdate • **ReportID** is a candidate key and Primary key which uniquely identifies all the attributes • Same UserID is on multiple Reports so not unique to relationship • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
10	 InteractionID=>interactinguserid InteractionID=>interactingplaceid InteractionID=>visitid	interactionid → interactinuserid interactionplaceid visitid • **InteractionID** is a candidate key and Primary key which uniquely identifies all the attributes • Same InteractingUseID, VisitorID and InteractionPlaceID can exist on multiple interactions so not unique to relationship • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
11	 eventid=>logbookid eventid=>description eventid=>eventstarttime eventid=>eventendtime	eventid → logbookid description starttime endtime • **EventID** is a candidate key and Primary key which uniquely identifies logbookid, description, eventstarttime, eventendtime • Same set of start time, end time and logbookid can exist in case the database is really large so they are not unique to relationship • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF

12	 placeid=>placename placeid=>streetarea placeid=>state placeid=>city placeid=>coountry placeid=>latitude placeid=>longitude	placeid → placename streetarea state city country latitude longitude • **placeid** is a candidate key and Primary key which uniquely identifies logbookid, description, eventstarttime, eventendtime • Same set of address and coordinates can exist in case for multiple persons sharing an address, so they are not unique to relationship • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
13		Nid,PersonID • (Nid,PersonID) is the only candidate key and a primary key, no non-prime attribute exist, so no dependency • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
14	 nid=>PHAauthorityid nid=>timestamp nid=>otherinformation	nid → PHAauthorityid timestamp otherinformation PHAauthorityid timestamp otherinformation → nid • **nid** is a candidate key and Primary key which uniquely identifies PHAAuthorityID, timestamp, otherinformation • trivial attributes (phaauthorityid, timestamp, otherinformation) form a secondary relationship. (Non-prime to key) • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF
15	 logid=>date logid=>assignedworkerid	logid → date assignedworkerid • **logid** is a candidate key and Primary key which uniquely identifies date and assignedworker • AssignedWorkerID can exist on multiple logs, so it is not unique to this relationship • No Partial Dependency or Multivalued Attribute exist in this relationship

		• There is no non-trivial FD without a candidate key. Hence table is in BCNF
16	 visitid=>visitinguserid visitid=>visitplaceid	visitid → visitplaceid visitinguserid • **VisitID** is a candidate key and Primary key which uniquely identifies VisitPlaceID and VisitingUserID • (VisitPlaceID,VisitingUserId) do not uniquely identify a visit because, there can be multiple visits, even on the same day • No Partial Dependency or Multivalued Attribute exist in this relationship • There is no non-trivial FD without a candidate key. Hence table is in BCNF

8. Integrity Checking Trials & Proofs

1.) Referential Integrity : Foreign Key Check

Test #1

Attempt to deleted a referenced record:

```
DELETE from AppUser where userid = 1499;
```

Response:

Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`contacttracingdb`.`appuseremail`, CONSTRAINT `appuseremail_ibfk_1` FOREIGN KEY (`UserID`) REFERENCES `appuser` (`UserID`)) 0.0039 sec

Test#2

Attempt to insert an unreferenced record:

```
INSERT INTO PublicHealthWorker
VALUES(1600,'MisterA',131245,13112);
```

Response:

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails (`contacttracingdb`.`publichealthworker`, CONSTRAINT `publichealthworker_ibfk_2` FOREIGN KEY (`WorkerID`) REFERENCES `person` (`PersonID`))

2.) Domain Integrity : Alien Value Check

Test #1

Attempt to insert a non-allowed value:

```
INSERT INTO Assessment VALUES(3899,1600,0,'2020-05-29 08:37:55','a random description','Friend',1511)
```

Response:

```
Error Code: 1265. Data truncated for column 'AssesseeType' at row 1
0.00028 sec
```

Test#2

Attempt to INSERT a value outside bounds:

```
INSERT INTO Place VALUES(1002,'Parua','RODQ PLACE','EAST BOSTON','MA','2128',42.36443246,-731234568910);
```

Response:

```
Error Code: 1264. Out of range value for column 'longitude' at row 1
0.00026 sec
```

3.) Entity Integrity and Key Constraints : Primary Key Null check

Test #1

Attempt to INSERT a NULL value for PRIMARY KEY :

```
INSERT INTO AppUserEmail(UserID, EmailID) VALUES (1101,null);
```

Response:

```
Error Code: 1048. Column 'EmailID' cannot be null
```

Test#2

Attempt to INSERT a VALUE in AUTO_INCREMENT Primary key field: We can clearly see that the Primary Key was autogenerated when a NULL value was passed.

```
INSERT INTO Place VALUES(NULL,'Aman','Batra
PLACE','SOUTH BOSTON','MA','2128',42.36467840,-
72.03322720);
Select * FROM PLACE where placename='Aman';
```

Response:

```
1 row(s) affected.

1001 Aman Batra PLACE    SOUTH BOSTON   MA   2128
     42.36467840    -72.03322720
```

Test#3: Checking Uniqueness of Keys
Counting primary keys of Places

```
SELECT placeid,COUNT(*) as total FROM place GROUP BY
placeid HAVING total > 1;
```

Response:

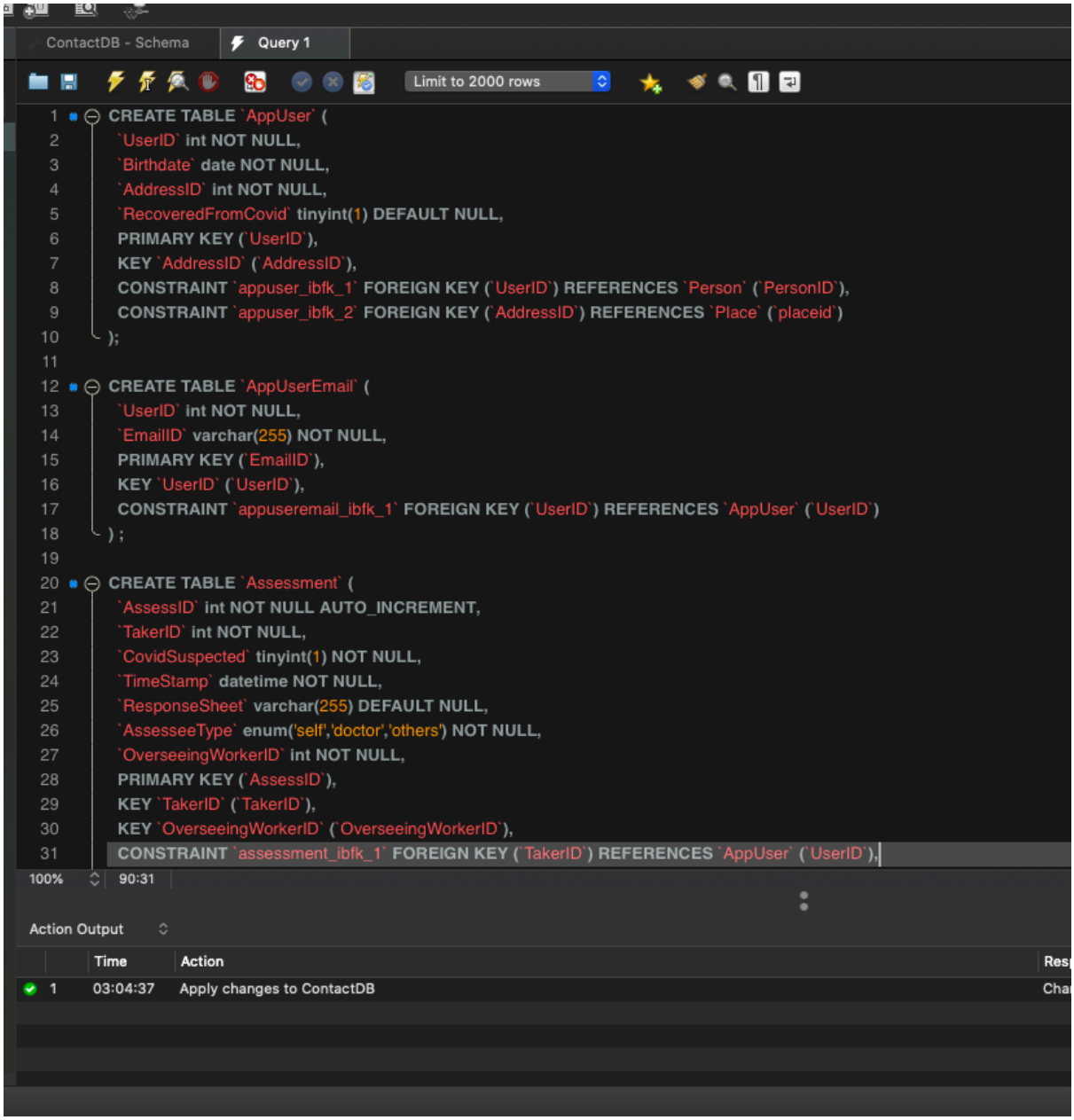
```
0 row(s) returned
```

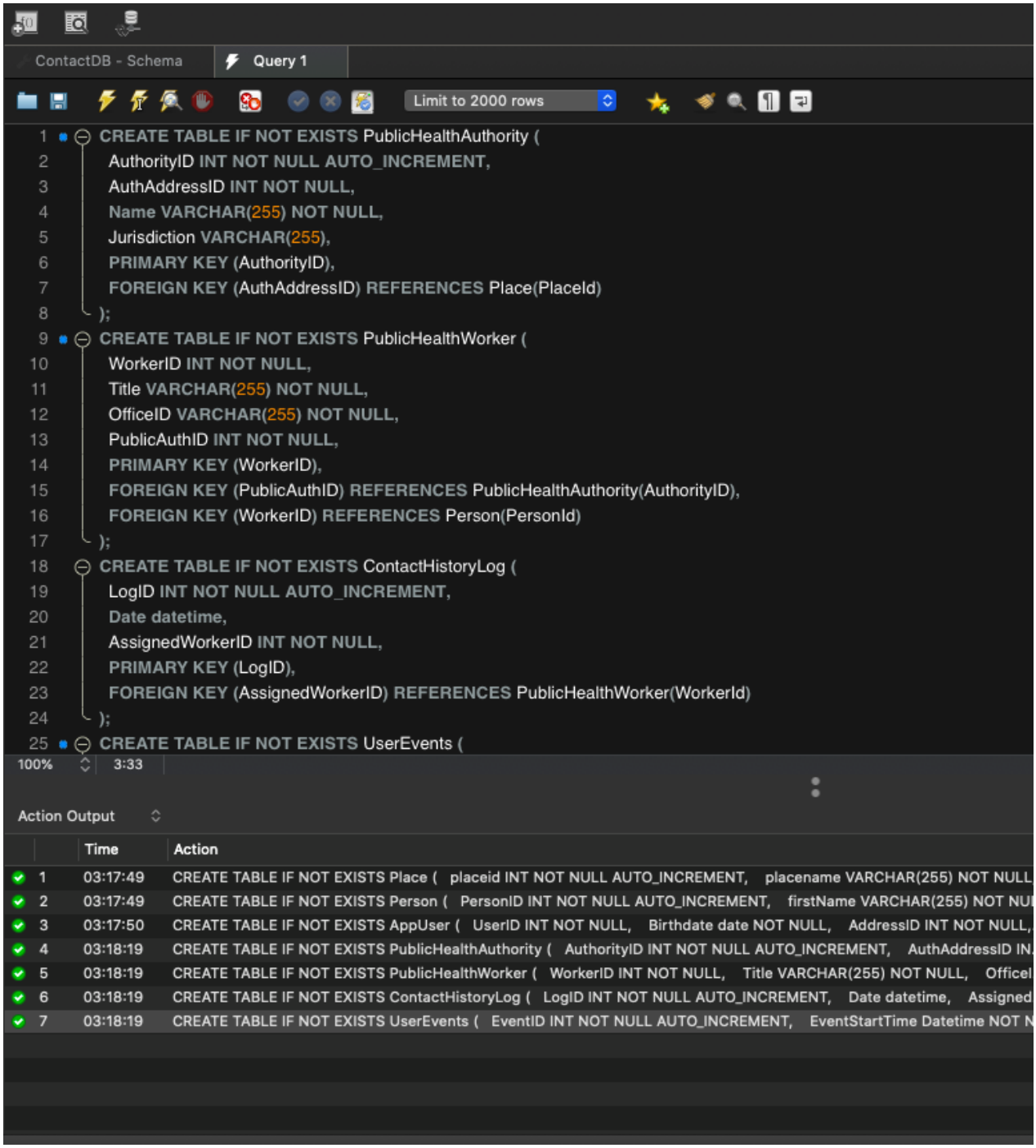
9. Create Tables & Insert Values

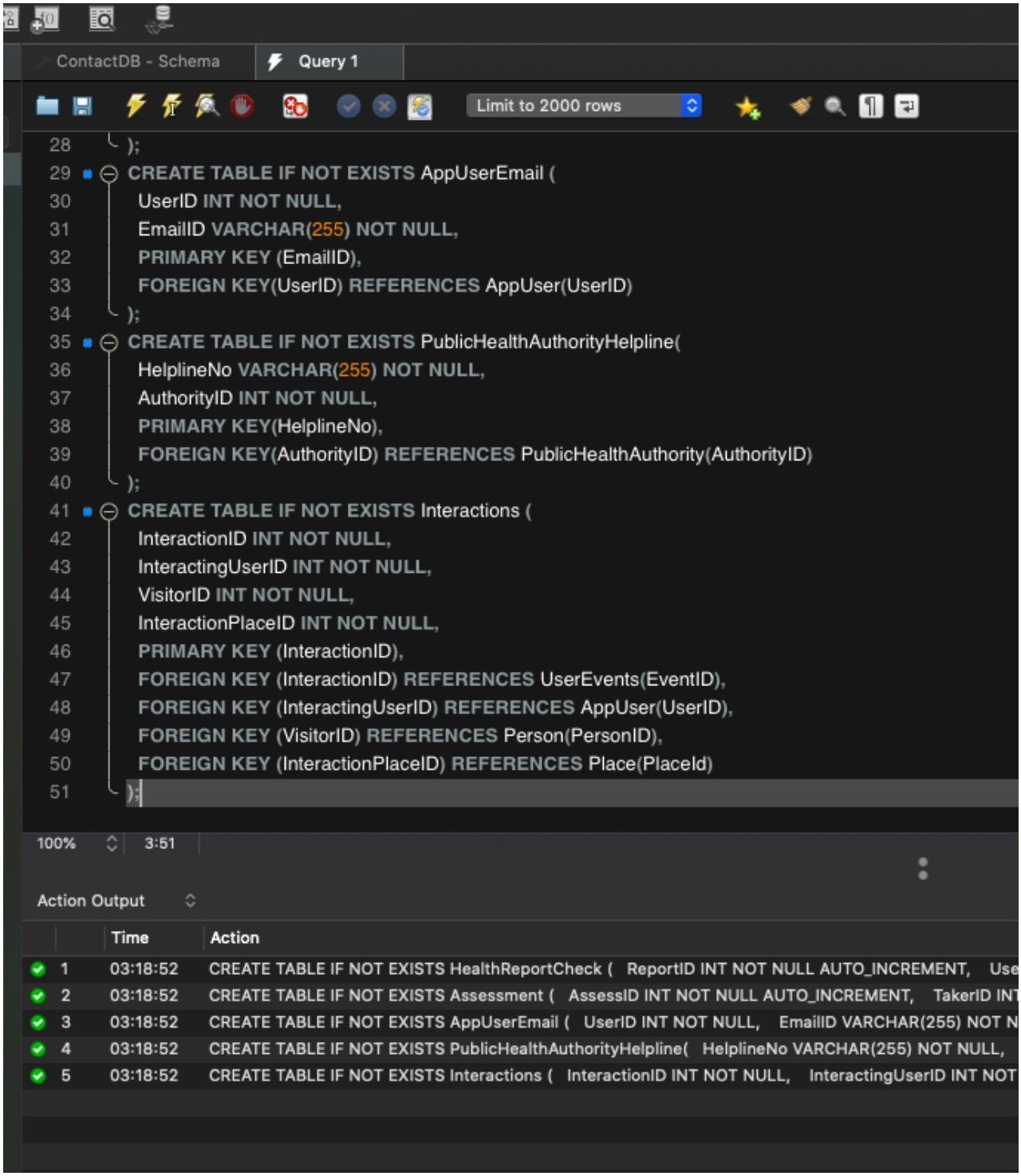
```
CREATE TABLE IF NOT EXISTS Place (
  placeid INT NOT NULL AUTO_INCREMENT,
  placename VARCHAR(255) NOT NULL,
  streetarea VARCHAR(255) NOT NULL,
  state VARCHAR(255) NOT NULL,
  city VARCHAR(255) NOT NULL,
  zip VARCHAR(255) NOT NULL,
  latitude decimal(10,8) signed NOT NULL,
  longitude decimal(11,8) signed NOT NULL,
  PRIMARY KEY (placeid)
);
CREATE TABLE IF NOT EXISTS Person (
  PersonID INT NOT NULL AUTO_INCREMENT,
  firstName VARCHAR(255) NOT NULL,
  lastName VARCHAR(255),
  PRIMARY KEY (PersonID)
);
CREATE TABLE IF NOT EXISTS AppUser (
  UserID INT NOT NULL,
  Birthdate date NOT NULL,
  AddressID INT NOT NULL,
  RecoveredFromCovid Boolean,
  PRIMARY KEY (UserID),
  FOREIGN KEY(UserID) REFERENCES Person(PersonID),
  FOREIGN KEY (AddressID) REFERENCES Place(PlaceID)
);
CREATE TABLE IF NOT EXISTS PublicHealthAuthority (
  AuthorityID INT NOT NULL AUTO_INCREMENT,
  AuthAddressID INT NOT NULL,
  Name VARCHAR(255) NOT NULL,
  Jurisdiction VARCHAR(255),
  PRIMARY KEY (AuthorityID),
  FOREIGN KEY (AuthAddressID) REFERENCES Place(PlaceId)
);
CREATE TABLE IF NOT EXISTS PublicHealthWorker (
  WorkerID INT NOT NULL,
  Title VARCHAR(255) NOT NULL,
  OfficeID VARCHAR(255) NOT NULL,
  PublicAuthID INT NOT NULL,
  PRIMARY KEY (WorkerID),
  FOREIGN KEY (PublicAuthID) REFERENCES
PublicHealthAuthority(AuthorityID),
  FOREIGN KEY (WorkerID) REFERENCES Person(PersonId)
);
CREATE TABLE IF NOT EXISTS ContactHistoryLog (
  LogID INT NOT NULL AUTO_INCREMENT,
  Date datetime,
  AssignedWorkerID INT NOT NULL,
  PRIMARY KEY (LogID),
  FOREIGN KEY (AssignedWorkerID) REFERENCES PublicHealthWorker(WorkerId)
);
```

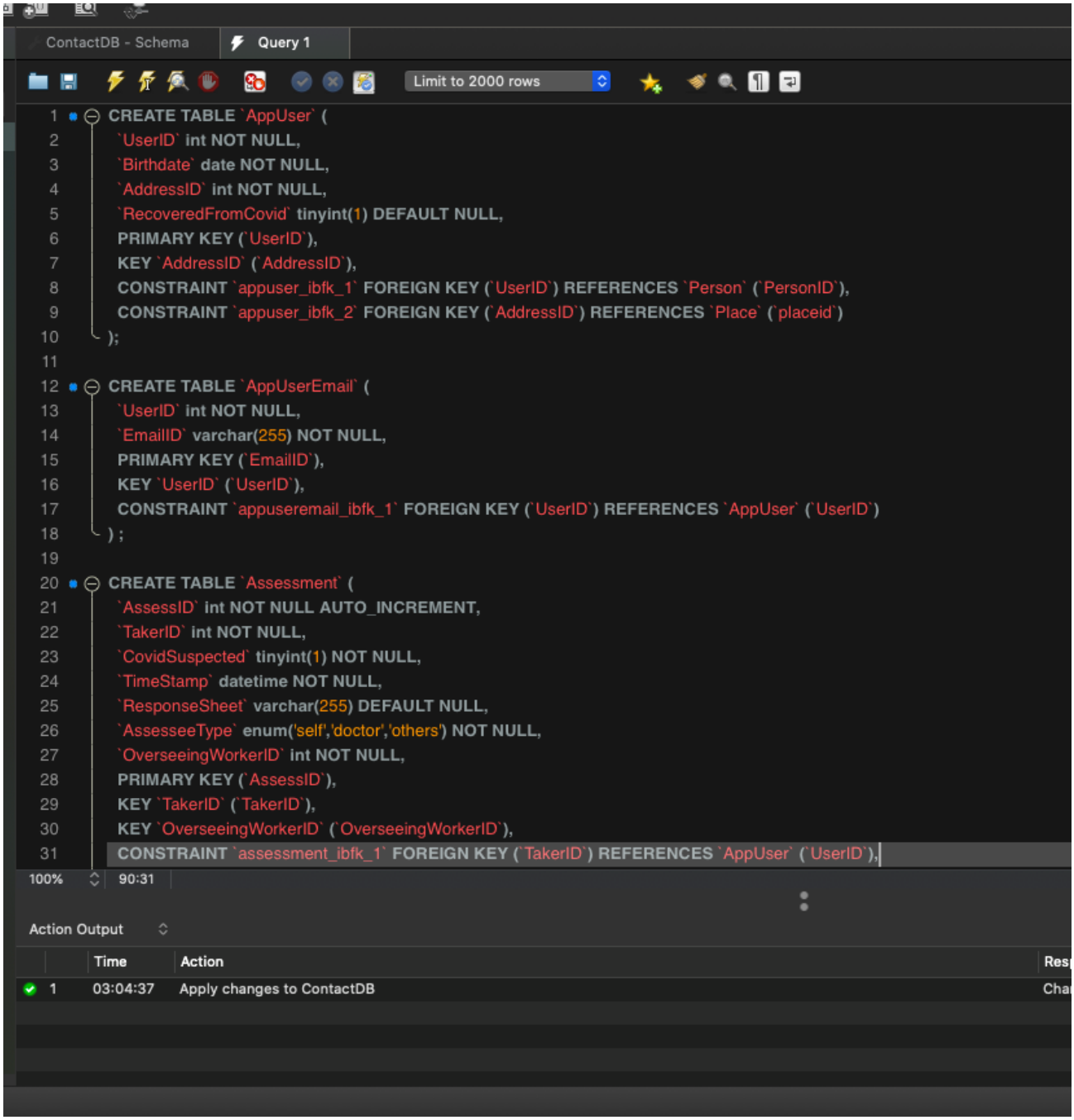
```
CREATE TABLE IF NOT EXISTS UserEvents (
  EventID INT NOT NULL AUTO_INCREMENT,
  EventStartTime Datetime NOT NULL,
  EventEndTime Datetime NOT NULL,
  LogBookID INT NOT NULL,
  Description VARCHAR(1200),
  PRIMARY KEY (EventID),
  FOREIGN KEY (LogBookID) REFERENCES ContactHistoryLog(LogID)
);
CREATE TABLE IF NOT EXISTS HealthReportCheck (
  ReportID INT NOT NULL AUTO_INCREMENT,
  UserID INT NOT NULL,
  Temperature decimal(10, 3),
  OxygenLevel decimal(10, 3),
  HealthStatus enum('sick', 'healthy') NOT NULL,
  TestResult enum(
    'Positive', 'Negative', 'Unclear'
  ) NOT NULL,
  Description Text,
  CheckingWorkerID INT NOT NULL,
  ReportDate date,
  PRIMARY KEY (ReportID),
  FOREIGN KEY (UserID) REFERENCES AppUser(UserID),
  FOREIGN KEY(CheckingWorkerID) REFERENCES PublicHealthWorker(WorkerId)
);
CREATE TABLE IF NOT EXISTS Assessment (
  AssessID INT NOT NULL AUTO_INCREMENT,
  TakerID INT NOT NULL,
  CovidSuspected boolean NOT NULL,
  TimeStamp Datetime NOT NULL,
  ResponseSheet VARCHAR(255),
  AssesseeType enum('self', 'doctor', 'others') NOT NULL,
  OverseeingWorkerID INT NOT NULL,
  PRIMARY KEY (AssessID),
  FOREIGN KEY (TakerID) REFERENCES AppUser(UserId),
  FOREIGN KEY (OverseeingWorkerID) REFERENCES
PublicHealthWorker(WorkerId)
);
CREATE TABLE IF NOT EXISTS AppUserEmail (
  UserID INT NOT NULL,
  EmailID VARCHAR(255) NOT NULL,
  PRIMARY KEY (EmailID),
  FOREIGN KEY(UserID) REFERENCES AppUser(UserID)
);
CREATE TABLE IF NOT EXISTS PublicHealthAuthorityHelpline(
  HelplineNo VARCHAR(255) NOT NULL,
  AuthorityID INT NOT NULL,
  PRIMARY KEY(HelplineNo),
  FOREIGN KEY(AuthorityID) REFERENCES PublicHealthAuthority(AuthorityID)
);
CREATE TABLE IF NOT EXISTS Interactions (
  InteractionID INT NOT NULL,
  InteractingUserID INT NOT NULL,
  VisitorID INT NOT NULL,
  InteractionPlaceID INT NOT NULL,
  PRIMARY KEY (InteractionID),
  FOREIGN KEY (InteractionID) REFERENCES UserEvents(EventID),
  FOREIGN KEY (InteractingUserID) REFERENCES AppUser(UserID),
  FOREIGN KEY (VisitorID) REFERENCES Person(PersonID),
  FOREIGN KEY (InteractionPlaceID) REFERENCES Place(PlaceId)
);
```

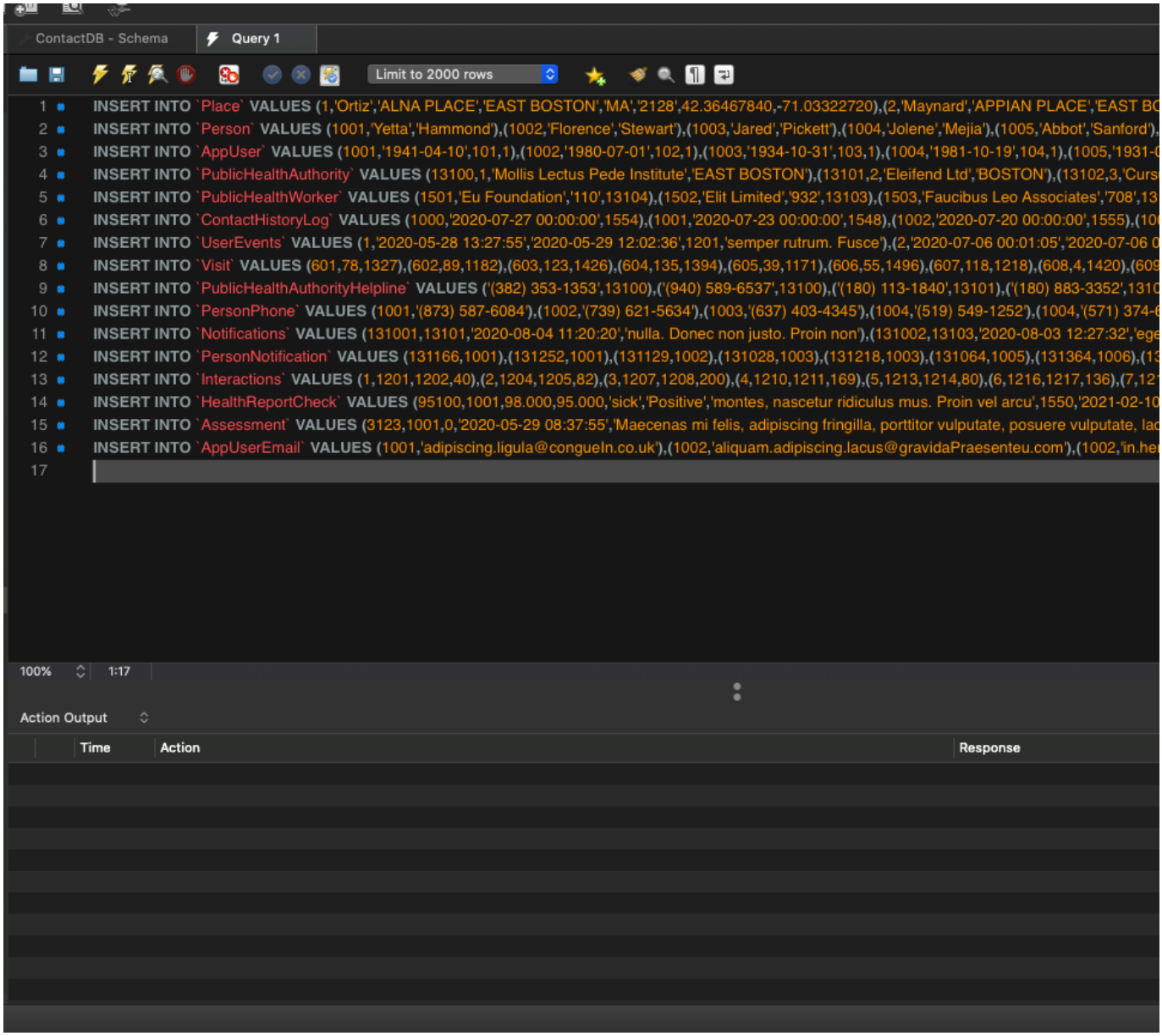
```
CREATE TABLE IF NOT EXISTS Visit (
  VisitID INT NOT NULL,
  VisitPlaceID INT NOT NULL,
  VisitingUserID INT NOT NULL,
  PRIMARY KEY (VisitID),
  FOREIGN KEY (VisitID) REFERENCES UserEvents(EventID),
  FOREIGN KEY (VisitPlaceID) REFERENCES Place(PlaceID),
  FOREIGN KEY(VisitingUserID) REFERENCES AppUser(UserID)
);
CREATE TABLE IF NOT EXISTS Notifications (
  Nid INT NOT NULL AUTO_INCREMENT,
  PHAAuthorityID INT NOT NULL,
  timeStamp datetime NOT NULL,
  OtherInformation Text,
  PRIMARY KEY (Nid),
  FOREIGN KEY (PHAAuthorityID) REFERENCES
PublicHealthAuthority(AuthorityId)
);
CREATE TABLE IF NOT EXISTS PersonNotification (
  Nid INT NOT NULL,
  PersonID INT NOT NULL,
  PRIMARY KEY (Nid, PersonID),
  FOREIGN KEY(Nid) REFERENCES Notifications(Nid),
  FOREIGN KEY(PersonID) REFERENCES Person(PersonID)
);
CREATE TABLE IF NOT EXISTS PersonPhone(
  PersonID INT NOT NULL,
  PhoneNo VARCHAR(20) NOT NULL,
  PRIMARY KEY (PhoneNo),
  FOREIGN KEY (PersonID) REFERENCES Person(PersonID)
);
```

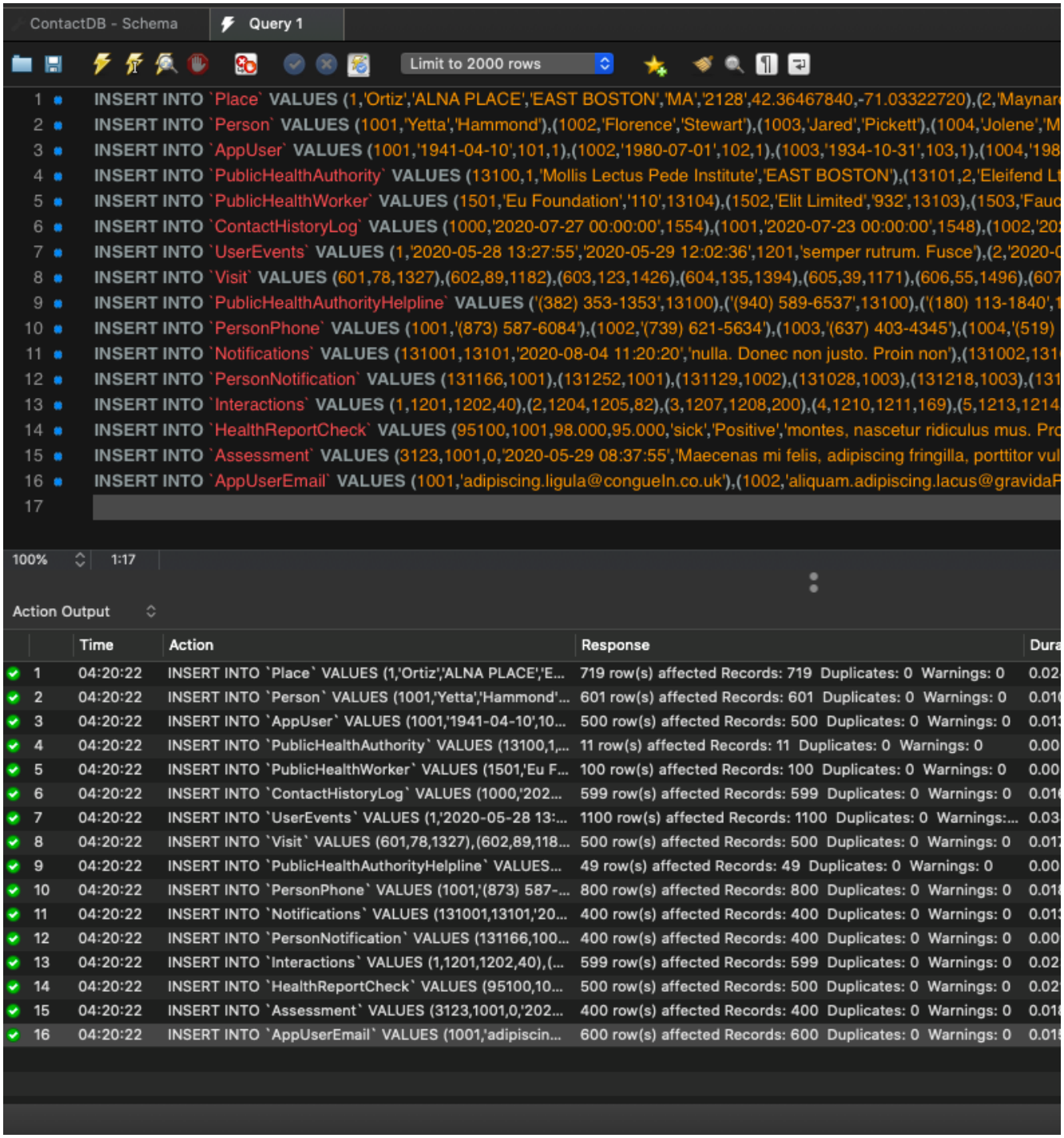

ContactDB - Schema
Query 1
Limit to 2000 rows
CREATE TABLE `AppUser` (
`UserID` int NOT NULL,
`Birthdate` date NOT NULL,
`AddressID` int NOT NULL,
`RecoveredFromCovid` tinyint(1) DEFAULT NULL,
PRIMARY KEY (`UserID`),
KEY `AddressID` (`AddressID`),
CONSTRAINT `appuser_ibfk_1` FOREIGN KEY (`UserID`) REFERENCES `Person` (`PersonID`),
CONSTRAINT `appuser_ibfk_2` FOREIGN KEY (`AddressID`) REFERENCES `Place` (`placeid`)
);
CREATE TABLE `AppUserEmail` (
`UserID` int NOT NULL,
`EmailID` varchar(255) NOT NULL,
PRIMARY KEY (`EmailID`),
KEY `UserID` (`UserID`),
CONSTRAINT `appuseremail_ibfk_1` FOREIGN KEY (`UserID`) REFERENCES `AppUser` (`UserID`)
) ;
CREATE TABLE `Assessment` (
`AssessID` int NOT NULL AUTO_INCREMENT,
`TakerID` int NOT NULL,
`CovidSuspected` tinyint(1) NOT NULL,
`TimeStamp` datetime NOT NULL,
`ResponseSheet` varchar(255) DEFAULT NULL,
`AssesseeType` enum('self','doctor','others') NOT NULL,
`OverseeingWorkerID` int NOT NULL,
PRIMARY KEY (`AssessID`),
KEY `TakerID` (`TakerID`),
KEY `OverseeingWorkerID` (`OverseeingWorkerID`),
CONSTRAINT `assessment_ibfk_1` FOREIGN KEY (`TakerID`) REFERENCES `AppUser` (`UserID`),
100%
90:31
Action Output
Time
Action
1
03:04:37
Apply changes to ContactDB


ContactDB - Schema
Query 1
Limit to 2000 rows
1 CREATE TABLE IF NOT EXISTS PublicHealthAuthority (
2 AuthorityID INT NOT NULL AUTO_INCREMENT,
3 AuthAddressID INT NOT NULL,
4 Name VARCHAR(255) NOT NULL,
5 Jurisdiction VARCHAR(255),
6 PRIMARY KEY (AuthorityID),
7 FOREIGN KEY (AuthAddressID) REFERENCES Place(PlaceId)
8);
9 CREATE TABLE IF NOT EXISTS PublicHealthWorker (
10 WorkerID INT NOT NULL,
11 Title VARCHAR(255) NOT NULL,
12 OfficeID VARCHAR(255) NOT NULL,
13 PublicAuthID INT NOT NULL,
14 PRIMARY KEY (WorkerID),
15 FOREIGN KEY (PublicAuthID) REFERENCES PublicHealthAuthority(AuthorityID),
16 FOREIGN KEY (WorkerID) REFERENCES Person(PersonId)
17);
18 CREATE TABLE IF NOT EXISTS ContactHistoryLog (
19 LogID INT NOT NULL AUTO_INCREMENT,
20 Date datetime,
21 AssignedWorkerID INT NOT NULL,
22 PRIMARY KEY (LogID),
23 FOREIGN KEY (AssignedWorkerID) REFERENCES PublicHealthWorker(WorkerId)
24);
25 CREATE TABLE IF NOT EXISTS UserEvents (
100%
3:33
Action Output
Time
Action
1 03:17:49 CREATE TABLE IF NOT EXISTS Place (placeid INT NOT NULL AUTO_INCREMENT, placename VARCHAR(255) NOT NULL
2 03:17:49 CREATE TABLE IF NOT EXISTS Person (PersonID INT NOT NULL AUTO_INCREMENT, firstName VARCHAR(255) NOT NU
3 03:17:50 CREATE TABLE IF NOT EXISTS AppUser (UserID INT NOT NULL, Birthdate date NOT NULL, AddressID INT NOT NULL,
4 03:18:19 CREATE TABLE IF NOT EXISTS PublicHealthAuthority (AuthorityID INT NOT NULL AUTO_INCREMENT, AuthAddressID IN
5 03:18:19 CREATE TABLE IF NOT EXISTS PublicHealthWorker (WorkerID INT NOT NULL, Title VARCHAR(255) NOT NULL, Officel
6 03:18:19 CREATE TABLE IF NOT EXISTS ContactHistoryLog (LogID INT NOT NULL AUTO_INCREMENT, Date datetime, Assigned
7 03:18:19 CREATE TABLE IF NOT EXISTS UserEvents (EventID INT NOT NULL AUTO_INCREMENT, EventStartTime Datetime NOT N


ContactDB - Schema
Query 1
Limit to 2000 rows
28);
29 CREATE TABLE IF NOT EXISTS AppUserEmail (
30 UserID INT NOT NULL,
31 EmailID VARCHAR(255) NOT NULL,
32 PRIMARY KEY (EmailID),
33 FOREIGN KEY(UserID) REFERENCES AppUser(UserID)
34);
35 CREATE TABLE IF NOT EXISTS PublicHealthAuthorityHelpline(
36 HelplineNo VARCHAR(255) NOT NULL,
37 AuthorityID INT NOT NULL,
38 PRIMARY KEY(HelplineNo),
39 FOREIGN KEY(AuthorityID) REFERENCES PublicHealthAuthority(AuthorityID)
40);
41 CREATE TABLE IF NOT EXISTS Interactions (
42 InteractionID INT NOT NULL,
43 InteractingUserID INT NOT NULL,
44 VisitorID INT NOT NULL,
45 InteractionPlaceID INT NOT NULL,
46 PRIMARY KEY (InteractionID),
47 FOREIGN KEY (InteractionID) REFERENCES UserEvents(EventID),
48 FOREIGN KEY (InteractingUserID) REFERENCES AppUser(UserID),
49 FOREIGN KEY (VisitorID) REFERENCES Person(PersonID),
50 FOREIGN KEY (InteractionPlaceID) REFERENCES Place(PlaceId)
51);
100%
3:51
Action Output
Time
Action
1 03:18:52 CREATE TABLE IF NOT EXISTS HealthReportCheck (ReportID INT NOT NULL AUTO_INCREMENT, Use
2 03:18:52 CREATE TABLE IF NOT EXISTS Assessment (AssessID INT NOT NULL AUTO_INCREMENT, TakerID INT
3 03:18:52 CREATE TABLE IF NOT EXISTS AppUserEmail (UserID INT NOT NULL, EmailID VARCHAR(255) NOT N
4 03:18:52 CREATE TABLE IF NOT EXISTS PublicHealthAuthorityHelpline(HelplineNo VARCHAR(255) NOT NULL,
5 03:18:52 CREATE TABLE IF NOT EXISTS Interactions (InteractionID INT NOT NULL, InteractingUserID INT NOT


ContactDB - Schema
Query 1
Limit to 2000 rows
CREATE TABLE `AppUser` (
`UserID` int NOT NULL,
`Birthdate` date NOT NULL,
`AddressID` int NOT NULL,
`RecoveredFromCovid` tinyint(1) DEFAULT NULL,
PRIMARY KEY (`UserID`),
KEY `AddressID` (`AddressID`),
CONSTRAINT `appuser_ibfk_1` FOREIGN KEY (`UserID`) REFERENCES `Person` (`PersonID`),
CONSTRAINT `appuser_ibfk_2` FOREIGN KEY (`AddressID`) REFERENCES `Place` (`placeid`)
);
CREATE TABLE `AppUserEmail` (
`UserID` int NOT NULL,
`EmailID` varchar(255) NOT NULL,
PRIMARY KEY (`EmailID`),
KEY `UserID` (`UserID`),
CONSTRAINT `appuseremail_ibfk_1` FOREIGN KEY (`UserID`) REFERENCES `AppUser` (`UserID`)
) ;
CREATE TABLE `Assessment` (
`AssessID` int NOT NULL AUTO_INCREMENT,
`TakerID` int NOT NULL,
`CovidSuspected` tinyint(1) NOT NULL,
`TimeStamp` datetime NOT NULL,
`ResponseSheet` varchar(255) DEFAULT NULL,
`AssesseeType` enum('self','doctor','others') NOT NULL,
`OverseeingWorkerID` int NOT NULL,
PRIMARY KEY (`AssessID`),
KEY `TakerID` (`TakerID`),
KEY `OverseeingWorkerID` (`OverseeingWorkerID`),
CONSTRAINT `assessment_ibfk_1` FOREIGN KEY (`TakerID`) REFERENCES `AppUser` (`UserID`),
100%
90:31
Action Output
Time
Action
1
03:04:37
Apply changes to ContactDB


ContactDB - Schema
Query 1
Limit to 2000 rows
1 INSERT INTO `Place` VALUES (1,'Ortiz','ALNA PLACE','EAST BOSTON','MA','2128',42.36467840,-71.03322720),(2,'Maynard','APPIAN PLACE','EAST BO
2 INSERT INTO `Person` VALUES (1001,'Yetta','Hammond'),(1002,'Florence','Stewart'),(1003,'Jared','Pickett'),(1004,'Jolene','Mejia'),(1005,'Abbot','Sanford'),
3 INSERT INTO `AppUser` VALUES (1001,'1941-04-10',101,1),(1002,'1980-07-01',102,1),(1003,'1934-10-31',103,1),(1004,'1981-10-19',104,1),(1005,'1931-0
4 INSERT INTO `PublicHealthAuthority` VALUES (13100,1,'Mollis Lectus Pede Institute','EAST BOSTON'),(13101,2,'Eleifend Ltd','BOSTON'),(13102,3,'Curs
5 INSERT INTO `PublicHealthWorker` VALUES (1501,'Eu Foundation','110',13104),(1502,'Elit Limited','932',13103),(1503,'Faucibus Leo Associates','708',13
6 INSERT INTO `ContactHistoryLog` VALUES (1000,'2020-07-27 00:00:00',1554),(1001,'2020-07-23 00:00:00',1548),(1002,'2020-07-20 00:00:00',1555),(10
7 INSERT INTO `UserEvents` VALUES (1,'2020-05-28 13:27:55','2020-05-29 12:02:36',1201,'semper rutrum. Fusce'),(2,'2020-07-06 00:01:05','2020-07-06 0
8 INSERT INTO `Visit` VALUES (601,78,1327),(602,89,1182),(603,123,1426),(604,135,1394),(605,39,1171),(606,55,1496),(607,118,1218),(608,4,1420),(609
9 INSERT INTO `PublicHealthAuthorityHelpline` VALUES ('(382) 353-1353',13100),('(940) 589-6537',13100),('(180) 113-1840',13101),('(180) 883-3352',1310
10 INSERT INTO `PersonPhone` VALUES (1001,'(873) 587-6084'),(1002,'(739) 621-5634'),(1003,'(637) 403-4345'),(1004,'(519) 549-1252'),(1004,'(571) 374-6
11 INSERT INTO `Notifications` VALUES (131001,13101,'2020-08-04 11:20:20','nulla. Donec non justo. Proin non'),(131002,13103,'2020-08-03 12:27:32','ege
12 INSERT INTO `PersonNotification` VALUES (131166,1001),(131252,1001),(131129,1002),(131028,1003),(131218,1003),(131064,1005),(131364,1006),(13
13 INSERT INTO `Interactions` VALUES (1,1201,1202,40),(2,1204,1205,82),(3,1207,1208,200),(4,1210,1211,169),(5,1213,1214,80),(6,1216,1217,136),(7,12
14 INSERT INTO `HealthReportCheck` VALUES (95100,1001,98.000,95.000,'sick','Positive','montes, nascetur ridiculus mus. Proin vel arcu',1550,'2021-02-10
15 INSERT INTO `Assessment` VALUES (3123,1001,0,'2020-05-29 08:37:55','Maecenas mi felis, adipiscing fringilla, porttitor vulputate, posuere vulputate, lac
16 INSERT INTO `AppUserEmail` VALUES (1001,'adipiscing.ligula@congueIn.co.uk'),(1002,'aliquam.adipiscing.lacus@gravidaPraesenteu.com'),(1002,'in.he
17
100%
1:17
Action Output
Time
Action
Response


ContactDB - Schema
Query 1
Limit to 2000 rows
1 INSERT INTO `Place` VALUES (1,'Ortiz','ALNA PLACE','EAST BOSTON','MA','2128',42.36467840,-71.03322720),(2,'Maynar
2 INSERT INTO `Person` VALUES (1001,'Yetta','Hammond'),(1002,'Florence','Stewart'),(1003,'Jared','Pickett'),(1004,'Jolene','M
3 INSERT INTO `AppUser` VALUES (1001,'1941-04-10',101,1),(1002,'1980-07-01',102,1),(1003,'1934-10-31',103,1),(1004,'198
4 INSERT INTO `PublicHealthAuthority` VALUES (13100,1,'Mollis Lectus Pede Institute','EAST BOSTON'),(13101,2,'Eleifend L
5 INSERT INTO `PublicHealthWorker` VALUES (1501,'Eu Foundation','110',13104),(1502,'Elit Limited','932',13103),(1503,'Fauc
6 INSERT INTO `ContactHistoryLog` VALUES (1000,'2020-07-27 00:00:00',1554),(1001,'2020-07-23 00:00:00',1548),(1002,'20
7 INSERT INTO `UserEvents` VALUES (1,'2020-05-28 13:27:55','2020-05-29 12:02:36',1201,'semper rutrum. Fusce'),(2,'2020-0
8 INSERT INTO `Visit` VALUES (601,78,1327),(602,89,1182),(603,123,1426),(604,135,1394),(605,39,1171),(606,55,1496),(607
9 INSERT INTO `PublicHealthAuthorityHelpline` VALUES ('(382) 353-1353',13100),('(940) 589-6537',13100),('(180) 113-1840',1
10 INSERT INTO `PersonPhone` VALUES (1001,'(873) 587-6084'),(1002,'(739) 621-5634'),(1003,'(637) 403-4345'),(1004,'(519)
11 INSERT INTO `Notifications` VALUES (131001,13101,'2020-08-04 11:20:20','nulla. Donec non justo. Proin non'),(131002,131
12 INSERT INTO `PersonNotification` VALUES (131166,1001),(131252,1001),(131129,1002),(131028,1003),(131218,1003),(131
13 INSERT INTO `Interactions` VALUES (1,1201,1202,40),(2,1204,1205,82),(3,1207,1208,200),(4,1210,1211,169),(5,1213,1214
14 INSERT INTO `HealthReportCheck` VALUES (95100,1001,98.000,95.000,'sick','Positive','montes, nascetur ridiculus mus. Pro
15 INSERT INTO `Assessment` VALUES (3123,1001,0,'2020-05-29 08:37:55','Maecenas mi felis, adipiscing fringilla, porttitor vul
16 INSERT INTO `AppUserEmail` VALUES (1001,'adipiscing.ligula@congueIn.co.uk'),(1002,'aliquam.adipiscing.lacus@gravidaP
17
100%
1:17
Action Output
Time
Action
Response
Dura
1 04:20:22 INSERT INTO `Place` VALUES (1,'Ortiz','ALNA PLACE','E... 719 row(s) affected Records: 719 Duplicates: 0 Warnings: 0 0.02
2 04:20:22 INSERT INTO `Person` VALUES (1001,'Yetta','Hammond'... 601 row(s) affected Records: 601 Duplicates: 0 Warnings: 0 0.01
3 04:20:22 INSERT INTO `AppUser` VALUES (1001,'1941-04-10',10... 500 row(s) affected Records: 500 Duplicates: 0 Warnings: 0 0.01
4 04:20:22 INSERT INTO `PublicHealthAuthority` VALUES (13100,1,... 11 row(s) affected Records: 11 Duplicates: 0 Warnings: 0 0.00
5 04:20:22 INSERT INTO `PublicHealthWorker` VALUES (1501,'Eu F... 100 row(s) affected Records: 100 Duplicates: 0 Warnings: 0 0.00
6 04:20:22 INSERT INTO `ContactHistoryLog` VALUES (1000,'202... 599 row(s) affected Records: 599 Duplicates: 0 Warnings: 0 0.01
7 04:20:22 INSERT INTO `UserEvents` VALUES (1,'2020-05-28 13:... 1100 row(s) affected Records: 1100 Duplicates: 0 Warnings:... 0.03
8 04:20:22 INSERT INTO `Visit` VALUES (601,78,1327),(602,89,118... 500 row(s) affected Records: 500 Duplicates: 0 Warnings: 0 0.01
9 04:20:22 INSERT INTO `PublicHealthAuthorityHelpline` VALUES... 49 row(s) affected Records: 49 Duplicates: 0 Warnings: 0 0.00
10 04:20:22 INSERT INTO `PersonPhone` VALUES (1001,'(873) 587-... 800 row(s) affected Records: 800 Duplicates: 0 Warnings: 0 0.01
11 04:20:22 INSERT INTO `Notifications` VALUES (131001,13101,'20... 400 row(s) affected Records: 400 Duplicates: 0 Warnings: 0 0.01
12 04:20:22 INSERT INTO `PersonNotification` VALUES (131166,100... 400 row(s) affected Records: 400 Duplicates: 0 Warnings: 0 0.00
13 04:20:22 INSERT INTO `Interactions` VALUES (1,1201,1202,40),(... 599 row(s) affected Records: 599 Duplicates: 0 Warnings: 0 0.02
14 04:20:22 INSERT INTO `HealthReportCheck` VALUES (95100,10... 500 row(s) affected Records: 500 Duplicates: 0 Warnings: 0 0.02
15 04:20:22 INSERT INTO `Assessment` VALUES (3123,1001,0,'202... 400 row(s) affected Records: 400 Duplicates: 0 Warnings: 0 0.01
16 04:20:22 INSERT INTO `AppUserEmail` VALUES (1001,'adipiscin... 600 row(s) affected Records: 600 Duplicates: 0 Warnings: 0 0.01

10. Query Generation, Scripts & Outputs

QUERY – 1 – A SIMPLE JOIN QUERY
Joining Tables: AppUser, Person, AppUserEmail
Goal : To find all the details of a user who tested positive in COVID-19 Drug Test

SQL:

```
SELECT * FROM AppUser
INNER JOIN Person ON AppUser.UserID=Person.PersonID
INNER JOIN AppUserEmail on
AppUser.UserID=AppUserEmail.UserID
INNER JOIN Place on AppUser.AddressID=Place.placeid
INNER JOIN HealthReportCheck on
AppUser.UserId=HealthReportCheck.UserID
WHERE HealthReportCheck.TestResult='positive';
```

```
Response: 153 row(s) returned
```

Result Screenshots:

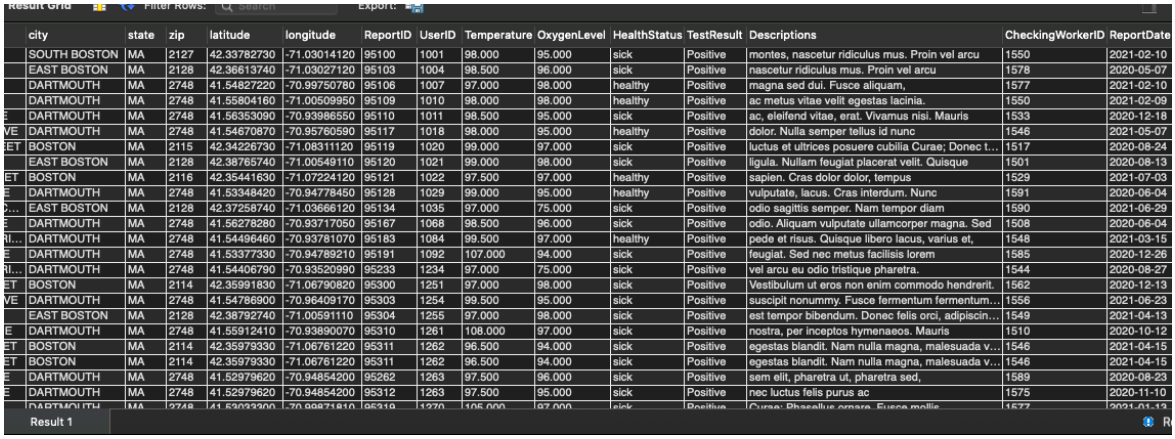
- Divided into two parts

PART 1

Result Grid Filter Rows: Search Export:

UserID	Birthdate	A...	...	PersonID	firstName	lastName	UserID	EmailID	placeid	placename	streetarea	city	state	zip	latitude	longitude
1001	1941-04-10	101	1	1001	Yetta	Hammond	1001	adipiscing.l...	101	Mclaughlin	ACADIA STREET	SOUTH BOSTON	MA	2127	42.33782730	-71.0301412
1004	1981-10-19	104	1	1004	Jolene	Mejia	1004	sapien@ne...	104	Santiago	ARDEE STREET	EAST BOSTON	MA	2128	42.36613740	-71.0302712
1007	1997-02-27	107	1	1007	Samuel	Flowers	1007	dapibus.id...	107	Blanda Unions	CEDAR AVENUE	DARTMOUTH	MA	2748	41.54827220	-70.9975078
1010	1973-12-17	110	1	1010	Petra	Bowers	1010	Mauris.ves...	110	Rogahn Manor	OLD JETTY ROAD	DARTMOUTH	MA	2748	41.55804160	-71.0050995
1011	2000-12-24	111	1	1011	Robin	Anderson	1011	sed.dui.Fu...	111	Scotty Alley	POKANOKET LANE	DARTMOUTH	MA	2748	41.56353090	-70.9398655
1018	1946-12-07	118	1	1018	Mercedes	Cruz	1018	et.malesua...	118	Sawayn Knoll	PLAINS FIELD DRIVE	DARTMOUTH	MA	2748	41.54670870	-70.9576059
1020	1968-09-06	120	1	1020	Tasha	Bush	1020	a.tortor.Nu...	120	Holder	ALBEMARLE STREET	BOSTON	MA	2115	42.34226730	-71.0831112
1021	1994-02-28	121	1	1021	Lucian	Klein	1021	lacinia.vita...	121	Slater	ANTRIM STREET	EAST BOSTON	MA	2128	42.38765740	-71.0054911
1022	2001-03-27	122	1	1022	Kevin	Yang	1022	a.nunc.In...	122	Conway	ARLINGTON STREET	BOSTON	MA	2116	42.35441630	-71.0722412
1029	1943-09-06	129	1	1029	Brendan	Price	1029	fringilla@di...	129	Ebert Divide	GOSNOLD AVENUE	DARTMOUTH	MA	2748	41.53348420	-70.9477845
1035	1976-12-25	135	1	1035	Kathleen	Garrett	1035	tincidunt.ali...	135	Clarke	ANTHONY J GRIEC...	EAST BOSTON	MA	2128	42.37258740	-71.0366612
1068	1943-04-19	168	1	1068	Alvin	Cash	1068	netus.et.m...	168	Shanahan Ford	POKANOKET LANE	DARTMOUTH	MA	2748	41.56278280	-70.9371705
1084	2006-09-11	184	1	1084	Elizabeth	Maloof	1084	ultricies.lig...	184	Nienow Turnp...	NORTH SHORE DRI...	DARTMOUTH	MA	2748	41.54496460	-70.9378107

PART 2



city	state	zip	latitude	longitude	ReportID	UserID	Temperature	OxygenLevel	HealthStatus	TestResult	Descriptions	CheckingWorkerID	ReportDate
SOUTH BOSTON	MA	2127	42.33782730	-71.03014120	95100	1001	98.000	95.000	sick	Positive	montes, nascetur ridiculus mus. Proin vel arcu	1550	2021-02-10
EAST BOSTON	MA	2128	42.36613740	-71.03027120	95103	1004	98.500	96.000	sick	Positive	nascetur ridiculus mus. Proin vel arcu	1578	2020-05-07
DARTMOUTH	MA	2748	41.54827220	-70.99750780	95106	1007	97.000	98.000	healthy	Positive	magna sed dui. Fusce aliquam,	1577	2021-02-10
DARTMOUTH	MA	2748	41.55804160	-71.00509950	95109	1010	98.000	98.000	healthy	Positive	ac metus vitae velit egestas lacinia.	1550	2021-02-09
DARTMOUTH	MA	2748	41.56353090	-70.93986550	95110	1011	98.500	95.000	sick	Positive	ac, eleifend vitae, erat. Vivamus nisi. Mauris	1533	2020-12-18
DARTMOUTH	MA	2748	41.54670870	-70.95760590	95117	1018	98.000	95.000	healthy	Positive	dolor. Nulla semper tellus id nunc	1546	2021-05-07
BOSTON	MA	2115	42.34226730	-71.08311120	95119	1020	99.000	97.000	sick	Positive	luctus et ultrices posuere cubilia Curae; Donec t...	1517	2020-08-24
EAST BOSTON	MA	2128	42.38765740	-71.00549110	95120	1021	99.000	98.000	sick	Positive	ligula. Nullam feugiat placerat velit. Quisque	1501	2020-08-13
BOSTON	MA	2116	42.35441630	-71.07224120	95121	1022	97.500	97.000	healthy	Positive	sapien. Cras dolor dolor, tempus	1529	2021-07-03
DARTMOUTH	MA	2748	41.53348420	-70.94778450	95128	1029	99.000	95.000	healthy	Positive	vulputate, lacus. Cras interdum. Nunc	1591	2020-06-04
EAST BOSTON	MA	2128	42.37258740	-71.03666120	95134	1035	97.000	75.000	sick	Positive	odio sagittis semper. Nam tempor diam	1590	2021-06-29
DARTMOUTH	MA	2748	41.56278280	-70.93717050	95167	1068	98.500	96.000	sick	Positive	odio. Aliquam vulputate ullamcorper magna. Sed	1508	2020-06-04
DARTMOUTH	MA	2748	41.54496460	-70.93781070	95183	1084	99.500	97.000	healthy	Positive	pede et risus. Quisque libero lacus, varius et,	1548	2021-03-15
DARTMOUTH	MA	2748	41.53377330	-70.94789210	95191	1092	107.000	94.000	sick	Positive	feugiat. Sed nec metus facilisis lorem	1585	2020-12-26
DARTMOUTH	MA	2748	41.54406790	-70.93520990	95233	1234	97.000	75.000	sick	Positive	vel arcu eu odio tristique pharetra.	1544	2020-08-27
BOSTON	MA	2114	42.35991830	-71.06790820	95300	1251	97.000	98.000	sick	Positive	Vestibulum ut eros non enim commodo hendrerit.	1562	2020-12-13
DARTMOUTH	MA	2748	41.54786900	-70.96409170	95303	1254	99.500	95.000	sick	Positive	suscipit nonummy. Fusce fermentum fermentum...	1556	2021-06-23
EAST BOSTON	MA	2128	42.38792740	-71.00591110	95304	1255	97.000	98.000	sick	Positive	est tempor bibendum. Donec felis orci, adipiscin...	1549	2021-04-13
DARTMOUTH	MA	2748	41.55912410	-70.93890070	95310	1261	108.000	97.000	sick	Positive	nostra, per inceptos hymenaeos. Mauris	1510	2020-10-12
BOSTON	MA	2114	42.35979330	-71.06761220	95311	1262	96.500	94.000	sick	Positive	egestas blandit. Nam nulla magna, malesuada v...	1546	2021-04-15
BOSTON	MA	2114	42.35979330	-71.06761220	95311	1262	96.500	94.000	sick	Positive	egestas blandit. Nam nulla magna, malesuada v...	1546	2021-04-15
DARTMOUTH	MA	2748	41.52979620	-70.94854200	95262	1263	97.500	96.000	sick	Positive	sem elit, pharetra ut, pharetra sed,	1589	2020-08-23
DARTMOUTH	MA	2748	41.52979620	-70.94854200	95312	1263	97.500	95.000	sick	Positive	nec luctus felis purus ac	1575	2020-11-10

Result 1

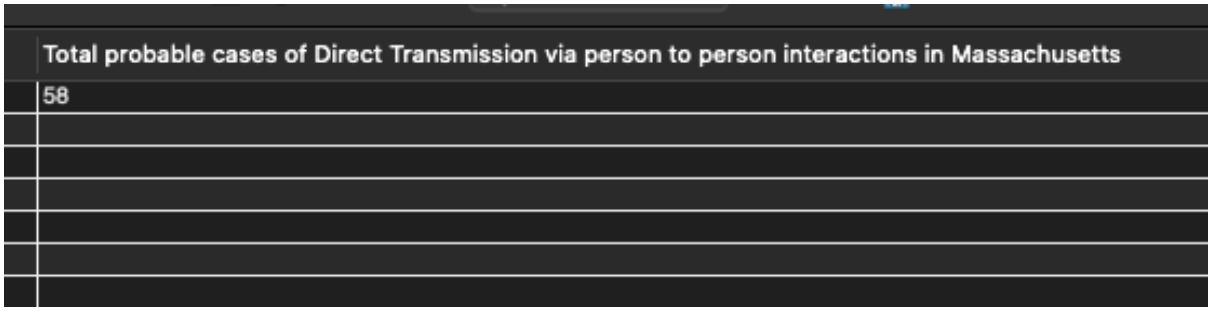
QUERY 2 – A Subquery to count total number of possible cases of direct person to person transmission in Massachusetts state.

SQL:

```
SELECT count(*) as  "Total probable cases of Direct
Transmission via person to person interactions in
Massachusetts"
FROM
 (
 SELECT i.InteractingUserID FROM Interactions i
INNER JOIN UserEvents on i.InteractionID=UserEvents.EventID
INNER JOIN HealthReportCheck r1 on
i.InteractingUserID=r1.UserID
INNER JOIN HealthReportCheck r2 on i.VisitorID=r2.UserID
INNER JOIN Place on  i.InteractionPlaceID=place.placeid
WHERE r1.TestResult='POSITIVE' and r2.TestResult in
('NEGATIVE','UNCLEAR')
AND r1.ReportDate<r2.ReportDate
AND Place.State in ('MA')
) as derived;
```

```
1 row(s) returned
```

Result Screenshots:



Total probable cases of Direct Transmission via person to person interactions in Massachusetts
58

QUERY 3 – A QUERY with a HAVING CLAUSE to return the details of visits made by people who tested positive, within 20 days(before and after) of getting the test reports. Where the visit lasted longer than 4 hours.

SQL:

```
SELECT a.UserID, p.firstName, s.PhoneNo, u.EventStartTime as
"Time of Visit",TIMEDIFF(u.EventEndTime,u.EventStartTime)as
DurationOfVisit, m.placename,m.streetarea, m.city,
m.longitude, m.latitude  FROM visit v
INNER JOIN AppUser a on v.VisitingUserID=a.UserID
INNER JOIN Person p on a.userid=p.PersonID
INNER JOIN PersonPhone s on p.PersonID=s.PersonID
INNER JOIN Place m on v.VisitPlaceID=m.placeid
INNER JOIN UserEvents u on v.VisitID=u.EventID
INNER JOIN HealthReportCheck h on a.UserID=h.UserID
WHERE h.TestResult='POSITIVE'
AND datediff(h.ReportDate, u.EventStartTime)<20
AND datediff(h.ReportDate, u.EventStartTime)>-20
HAVING DurationOfVisit>'04:00:00'
ORDER by DurationOfVisit
```

```
9 row(s) returned
```

RESULT SCREENSHOTS:

Result Grid Filter Rows: Search Export:

UserID	firstName	PhoneNo	Time of Visit	DurationOfVi...	placename	streetarea	city	longitude	latitude
1260	Aileen	(490) 564-4413	2020-07-11 20:13:24	04:50:49	Sawayn Knoll	PLAINS FIELD DRIVE	DARTMOUTH	-70.95760590	41.54670870
1271	Aquila	(347) 164-3418	2020-07-19 11:29:43	15:40:44	Kyra Plaza	MEADOW SHORES ROAD	DARTMOUTH	-70.94757720	41.53699490
1336	Isadora	(657) 930-9922	2020-07-14 03:18:15	18:50:07	Kuhic Brooks	MEADOW SHORES ROAD	DARTMOUTH	-70.94763740	41.53670050
1276	Patrick	(234) 147-8947	2020-07-03 22:31:00	19:24:46	Kuhic Brooks	MEADOW SHORES ROAD	DARTMOUTH	-70.94763740	41.53670050
1276	Patrick	(571) 689-5951	2020-07-03 22:31:00	19:24:46	Kuhic Brooks	MEADOW SHORES ROAD	DARTMOUTH	-70.94763740	41.53670050
1398	Simon	(187) 845-7356	2020-07-11 11:50:19	20:11:41	Wisoky Turnpike	WAMSUTTA STREET	DARTMOUTH	-70.93867720	41.55821490
1276	Patrick	(234) 147-8947	2020-07-18 06:00:38	20:35:01	Kyra Plaza	MEADOW SHORES ROAD	DARTMOUTH	-70.94757720	41.53699490
1276	Patrick	(571) 689-5951	2020-07-18 06:00:38	20:35:01	Kyra Plaza	MEADOW SHORES ROAD	DARTMOUTH	-70.94757720	41.53699490
1271	Aquila	(347) 164-3418	2020-07-09 13:13:58	23:16:16	Satterfield Flats	PENNIKESE LANE	DARTMOUTH	-70.93912920	41.55689690

QUERY 4 – A COMPLEX QUERY – A depth-2 search query to find out all interactions of persons, who came out positive/unclear in the covid-19 test, after their interacting with some person in the past.

Example Case:

- John Doe meets Sabrina Chan, and Jian Yang.
- John Doe was found positive, so Sabrina Chan and Jian Yang take covid tests.
- Jian tests positive, while Sabrina tests negative, so we need to trace Jian's interactions now.

SQL:

```
select o.FirstName as "Guy1 who found out he got exposed",
p.FirstName as "Guy2 who had met guy1", s.PhoneNo as "Phone
number of Guy2", u.eventstarttime as "Interaction Start
Details" , u.eventendtime as "Interaction End Details" from
interactions a, person p, personphone s,userevents u, person
o
WHERE a.InteractionID=u.eventid
AND p.personid = a.visitorid
AND s.PersonID=p.PersonId
AND a.Interactinguserid=o.personid

AND a.interactinguserid IN (
SELECT  i.VisitorID FROM Interactions i
INNER JOIN UserEvents on i.InteractionID=UserEvents.EventID
INNER JOIN HealthReportCheck r1 on
i.InteractingUserID=r1.UserID
INNER JOIN HealthReportCheck r2 on i.VisitorID=r2.UserID
WHERE r1.TestResult='POSITIVE' and r2.TestResult in
('UNCLEAR','POSITIVE')
AND  r1.ReportDate< r2.ReportDate
)

AND a.Interactinguserid NOT IN (
SELECT  i.interactinguserid FROM Interactions i
INNER JOIN UserEvents on i.InteractionID=UserEvents.EventID
INNER JOIN HealthReportCheck r1 on
i.InteractingUserID=r1.UserID
INNER JOIN HealthReportCheck r2 on i.VisitorID=r2.UserID
WHERE r1.TestResult='POSITIVE' and r2.TestResult in
('UNCLEAR','POSITIVE')
AND  r1.ReportDate< r2.ReportDate
)
```

```
37 row(s) returned
```

RESULT SCREENSHOTS

Result Grid Filter Rows: Search Export:

Guy1 who found out he got exposed	Guy2 who had met guy1	Phone number of Guy2	Interaction Start Details	Interaction End Details
Hyatt	Meredith	(879) 909-7095	2020-09-19 09:49:27	2020-09-19 16:56:19
Blake	Deacon	(577) 819-8645	2020-12-02 16:46:01	2020-12-02 18:32:43
Blake	Deacon	(577) 819-8645	2020-06-14 15:03:22	2020-06-15 13:43:53
Blake	Deacon	(577) 819-8645	2020-07-24 21:34:46	2020-07-24 23:40:24
Isabella	Nissim	(299) 210-1873	2020-07-01 09:49:14	2020-07-02 06:04:40
Jonah	Harrison	(855) 502-7133	2020-11-06 21:28:28	2020-11-07 03:22:17
Harding	Raphael	(329) 783-9231	2020-07-05 10:26:13	2020-07-05 17:27:59
Harding	Raphael	(556) 414-9216	2020-07-05 10:26:13	2020-07-05 17:27:59
Harding	Raphael	(718) 485-5570	2020-07-05 10:26:13	2020-07-05 17:27:59
Harding	Raphael	(329) 783-9231	2020-11-05 17:44:23	2020-11-06 16:42:42
Harding	Raphael	(556) 414-9216	2020-11-05 17:44:23	2020-11-06 16:42:42
Harding	Raphael	(718) 485-5570	2020-11-05 17:44:23	2020-11-06 16:42:42
Nissim	Eliana	(601) 906-0421	2020-02-20 13:30:40	2020-02-21 05:47:24
Nissim	Eliana	(624) 566-4597	2020-02-20 13:30:40	2020-02-21 05:47:24
Regan	Iris	(371) 322-2736	2020-02-22 05:18:29	2020-02-22 15:42:22
Regan	Iris	(407) 669-0388	2020-02-22 05:18:29	2020-02-22 15:42:22
Regan	Iris	(441) 434-8855	2020-02-22 05:18:29	2020-02-22 15:42:22
Jillian	Lucas	(459) 337-1561	2020-12-26 15:40:00	2020-12-27 05:10:50
Jillian	Lucas	(526) 380-9378	2020-12-26 15:40:00	2020-12-27 05:10:50
Jillian	Mia	(111) 440-6493	2020-06-05 23:56:21	2020-06-06 00:44:52

Result 64

Note: We don’t use the distinct keyword because we want to find out all interactions, even if there were more than one!

We can go till depth 5 with MySQL in a medium sized database, after depth-5 mysqld crashes within 30 minutes of wait.

QUERY 5 – A Query of Choice

Details of interactions of persons who tested COVID positive(Depth 1 -> Guy 0 meets Guy 1)

SQL:

```
SELECT i.interactingUserId as UniqueID, guy1.firstName as
"Meeting Person 1", r1.TestResult as "First Guy's Covid
Report"
,i.VisitorID as UniqueID,guy2.firstName as "Meeting Person
2", r2.TestResult as "Second Guy's Covid Report",
UserEvents.EventStartTime as "DateTime of Meeting",
UserEvents.Description as "Details of Meeting"
 FROM Interactions i
INNER JOIN UserEvents on i.InteractionID=UserEvents.EventID
INNER JOIN Person guy1 on i.InteractingUserID=guy1.PersonID
INNER JOIN Person guy2 on i.VisitorID=guy2.PersonID
INNER JOIN HealthReportCheck r1 on
i.InteractingUserID=r1.UserID
INNER JOIN HealthReportCheck r2 on i.VisitorID=r2.UserID
WHERE r1.TestResult='POSITIVE'
```

```
228 row(s) returned
```

RESULTS SCREENSHOTS:

Result Grid Filter Rows: Search Export:

UniqueID	Meeting Person 1	First Guy's Covid Report	UniqueID	Meeting Person 2	Second Guy's Covid Rep...	DateTime of Meeti...	Details of Meeting
1001	Yetta	Positive	1002	Florence	Unclear	2020-02-14 00:36:23	mi. Duis risus odio, auctor vitae, aliquet
1001	Yetta	Positive	1002	Florence	Unclear	2020-04-06 20:06:35	quam vel sapien imperdiet ornare. In faucibus.
1005	Abbot	Positive	1006	Abbot	Negative	2020-11-01 17:06:41	Mauris ut quam vel sapien imperdiet ornare.
1005	Abbot	Positive	1006	Abbot	Negative	2020-01-04 18:26:57	iaculis nec, eleifend
1005	Abbot	Positive	1006	Abbot	Negative	2020-04-20 02:11:17	lectus ante dictum mi, ac mattis
1007	Samuel	Positive	1008	Kyra	Positive	2020-05-07 20:01:34	et ultrices posuere cubilia Curae;
1010	Petra	Positive	1011	Robin	Positive	2020-03-16 14:35:58	enim, condimentum eget, volutpat
1011	Robin	Positive	1012	Malcolm	Positive	2020-02-27 16:54:27	pede. Suspendisse dui. Fusce
1020	Tasha	Positive	1021	Lucian	Positive	2020-04-06 12:07:44	sem elit, pharetra ut, pharetra sed,
1021	Lucian	Positive	1022	Kevin	Positive	2020-09-23 07:33:02	est ac mattis semper, dui lectus rutrum urna,
1021	Lucian	Positive	1022	Kevin	Positive	2020-04-15 09:32:10	Praesent eu dui.
1027	Plato	Positive	1028	Aurora	Unclear	2020-05-24 18:38:17	id risus quis diam luctus lobortis. Class
1029	Brendan	Positive	1030	Kirk	Negative	2020-05-23 06:12:18	massa. Integer vitae
1029	Brendan	Positive	1030	Kirk	Negative	2020-12-23 23:52:40	nec, mollis vitae, posuere
1033	Roanna	Positive	1034	Gray	Unclear	2020-11-18 19:24:50	Sed molestie. Sed id risus quis
1033	Roanna	Positive	1034	Gray	Unclear	2020-03-11 07:22:00	et malesuada fames ac
1035	Kathleen	Positive	1036	Echo	Negative	2020-04-04 19:37:43	enim nisi elementum

Result 71

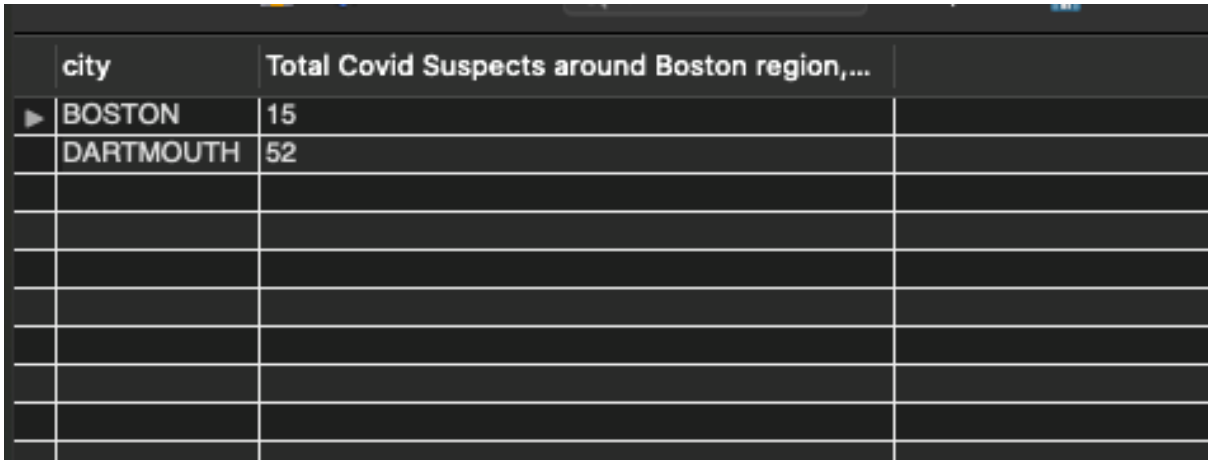
QUERY 6 – EXTRA QUERY

A QUERY with HAVING clause to count the no of people per city above 60 years in age, who took the Covid-19 Assessment survey, and suspected an infection, Only listing cities with more than 10 distinct suspects.

SQL:

```
SELECT distinct Place.city, count(distinct
Assessment.TakerID) as "Total Covid Suspects around Boston
region, with age greater than 50" FROM Place
INNER JOIN AppUser on Place.placeid=AppUser.addressid
INNER JOIN Assessment on AppUser.UserID=Assessment.TakerID
WHERE Assessment.CovidSuspected=1 AND
AppUser.Birthdate<'1960-01-01'
Group By Place.City HAVING count(distinct
Assessment.TakerID)>10;
```

Result Screenshots:



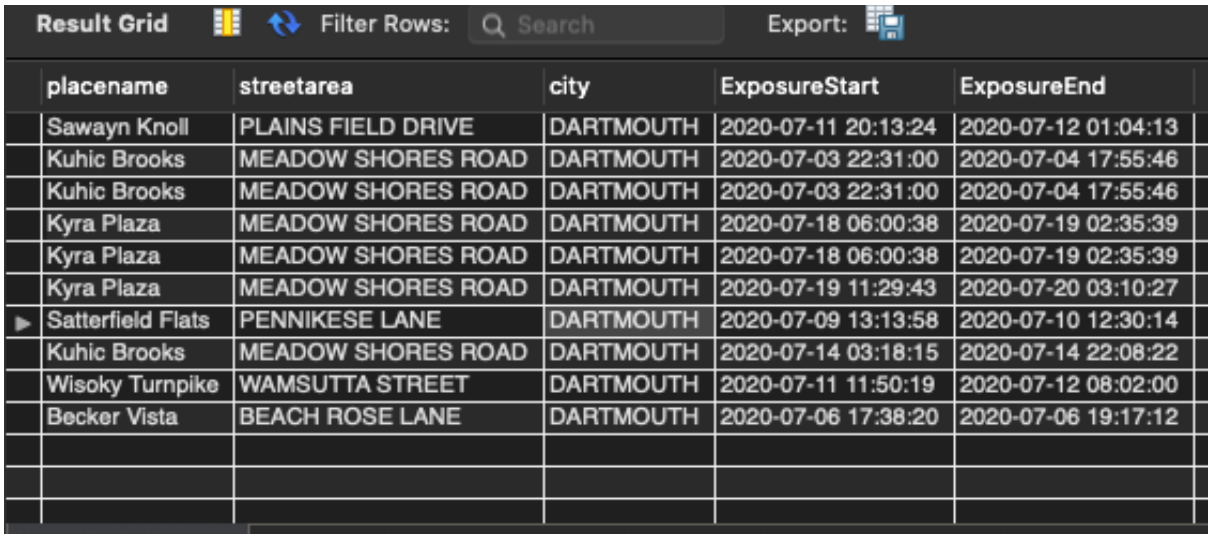
city	Total Covid Suspects around Boston region,...
BOSTON	15
DARTMOUTH	52

QUERY 7 – EXTRA QUERY
Exposure events and places and Massachusetts. This query backtracks the visit log of a COVID-19 positive persons and returns exact places where they visited and spent some time.

```
SELECT m.placename,streetarea, m.city,u.EventStartTime as
ExposureStart, u.EventEndTime as ExposureEnd FROM visit v
INNER JOIN AppUser a on v.VisitingUserID=a.UserID
INNER JOIN Person p on a.userid=p.PersonID
INNER JOIN PersonPhone s on p.PersonID=s.PersonID
INNER JOIN Place m on v.VisitPlaceID=m.placeid
INNER JOIN UserEvents u on v.VisitID=u.EventID
INNER JOIN HealthReportCheck h on a.UserID=h.UserID
WHERE h.TestResult='POSITIVE'
AND M.state='MA'
AND datediff(h.ReportDate, u.EventStartTime)<20
AND datediff(h.ReportDate, u.EventStartTime)>-20
```

```
10 row(s) returned
```

RESULT SCREENSHOTS:



placename	streetarea	city	ExposureStart	ExposureEnd
Sawayn Knoll	PLAINS FIELD DRIVE	DARTMOUTH	2020-07-11 20:13:24	2020-07-12 01:04:13
Kuhic Brooks	MEADOW SHORES ROAD	DARTMOUTH	2020-07-03 22:31:00	2020-07-04 17:55:46
Kuhic Brooks	MEADOW SHORES ROAD	DARTMOUTH	2020-07-03 22:31:00	2020-07-04 17:55:46
Kyra Plaza	MEADOW SHORES ROAD	DARTMOUTH	2020-07-18 06:00:38	2020-07-19 02:35:39
Kyra Plaza	MEADOW SHORES ROAD	DARTMOUTH	2020-07-18 06:00:38	2020-07-19 02:35:39
Kyra Plaza	MEADOW SHORES ROAD	DARTMOUTH	2020-07-19 11:29:43	2020-07-20 03:10:27
Satterfield Flats	PENNIKESE LANE	DARTMOUTH	2020-07-09 13:13:58	2020-07-10 12:30:14
Kuhic Brooks	MEADOW SHORES ROAD	DARTMOUTH	2020-07-14 03:18:15	2020-07-14 22:08:22
Wisoky Turnpike	WAMSUTTA STREET	DARTMOUTH	2020-07-11 11:50:19	2020-07-12 08:02:00
Becker Vista	BEACH ROSE LANE	DARTMOUTH	2020-07-06 17:38:20	2020-07-06 19:17:12

11. MySQL Integration with R-Studio, Integrated Queries and Plots