Data-driven Application Project

Phase 2

CSE 412 - Database Management

Andrew Dudley
Cameron Dudley
Andrew Stanton
Michael Heaton
Steven Nguyen

March 14, 2017

Criminal Justice Report Application overview

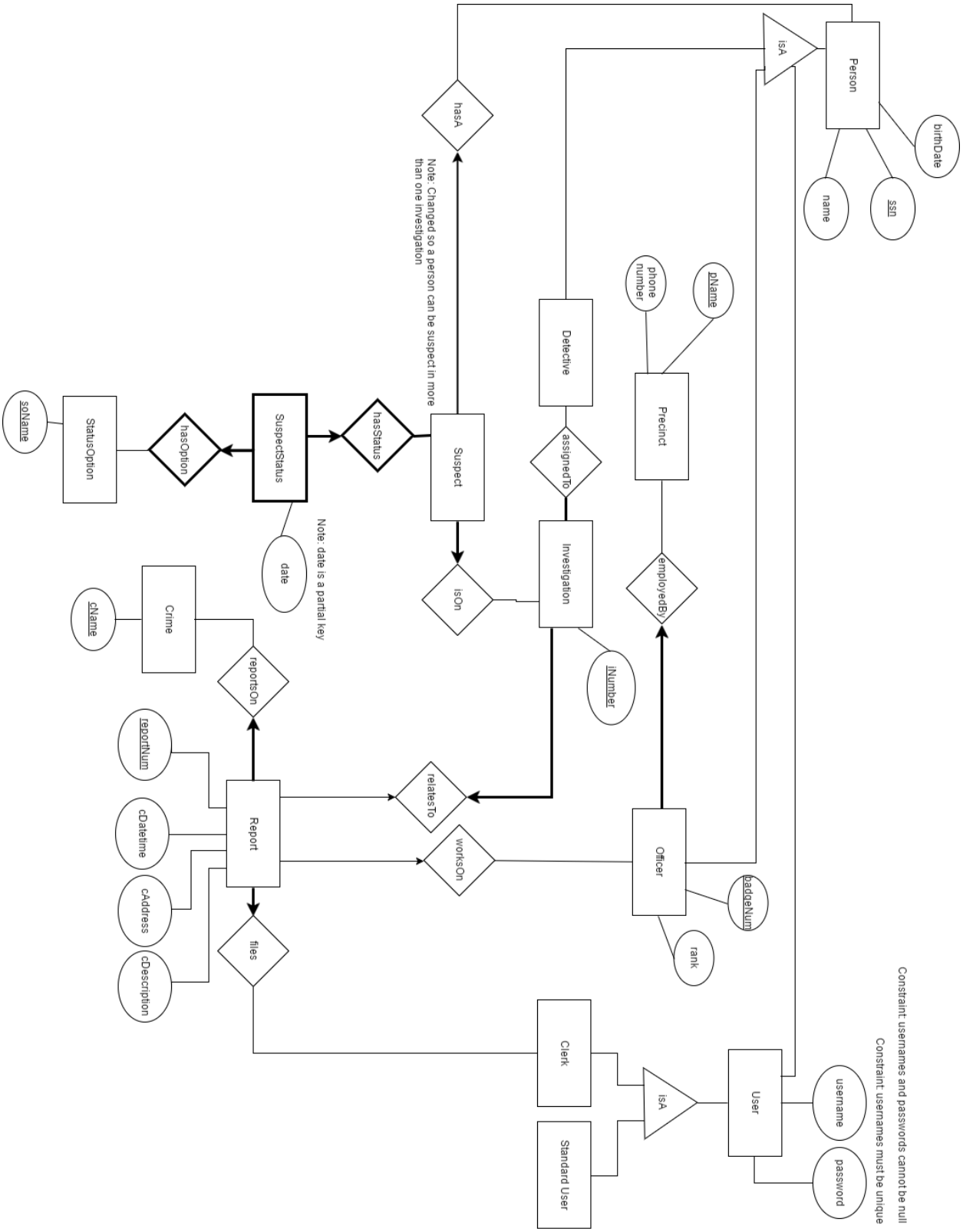
Database systems play a key role in the modern criminal justice system. Digital reports need to be generated for each incident, and this data must then be accessed by several departments throughout the lifecycle of a criminal report. Once the lifecycle has ended, these reports must still be archived and accessible for other users, such as data scientists wishing to perform data analytics on crime rates.

This application provides a CRUD implementation for criminal justice reports, enabling users with varying roles and permissions to search, create, and update criminal reports, as well as to access additional information about the individuals involved in the filing, investigation, arbitration, and closure of each case. The search functionality will enable researchers and data analysts to easily access and analyze this information.

For our application we are modeling a police and law enforcement database system. The police officers are to compile data for reports on crimes that occur in a particular precinct. The role of the clerk is to take the data on a crime that was gathered by the police officers and produce the report. Detectives then have the role of conducting an investigation on the people who are suspected of those crimes. Our database features to implement these functionalities are as follows:

- Clerks are able to file and store reports on the crimes that occur, which address they occur at, the time they occur, and a description of the incident.
- Each suspect has a 'status option' indicating their current status. These are: Arrested, in jail, under investigation, and under parole.
- Every police officer is employed by a particular precinct which they serve. This allows us to query which officers are employed by a precinct.
- Detectives are assigned to an investigation on a particular suspect. This allows us to query which detectives investigated a certain suspect.
- A standard user (perhaps a researcher or member of the public) can construct queries that allow them to view a report on a crime and a description of the crime that occurred.
- A standard user can use a query to retrieve data on which police officers apprehended a particular suspect. They can do this by using the report entity to link a police officer to a particular apprehension.
- A standard user can use a query to retrieve data on which officers compiled data for a report during a particular year.
- A standard user can use a query to find out which detectives have investigated the same suspect on multiple incidents.
- A standard user can use a query to find out which police precinct apprehended the most suspects on a particular year.

E/R Diagram


Person
isA
birthDate
name
ssn
hasA
Note: Changed so a person can be suspect in more than one investigation
Detective
phone number
pName
Precinct
assignedTo
employedBy
Suspect
hasStatus
SuspectStatus
hasOption
StatusOption
soName
date
Note: date is a partial key
Investigation
isOn
iNumber
Crime
cName
reportsOn
Report
reportNum
cDatetime
cAddress
cDescription
files
relatesTo
worksOn
Officer
badgeNum
rank
Clerk
Standard User
isA
User
username
password
Constraint: usernames and passwords cannot be null
Constraint usernames must be unique

E/R to SQL statements

The Entity Sets and their relationships illustrated in the E/R diagram are converted into SQL tables named after those entity sets and relationships, providing a straight-forward translation from the diagram to the resulting database. The scripts for the data definition language (DDL) for creating tables can be found in the 'DDL scripts.sql' file.

- Person (ES) - implemented by CREATE TABLE Person.
  Inheritance:
  - Officer (ES), employedBy (R) - implemented by 'CREATE TABLE officer_employedby'. Subsumption of Person to Officer entity sets are implemented using the PRIMARY(ssn) from Person table.
  - Detective (ES) - implemented by 'CREATE TABLE detective'. Subsumption of Person to Detective entity sets implemented using the PRIMARY(ssn) from Person table.
  - User (ES) - implemented by 'CREATE TABLE user'.
    Inheritance:
    * Standard User (ES) - implemented by 'CREATE TABLE standard_user'. Subsumption of User to Standard User entity sets implemented using the PRIMARY(ssn) from User table.
    * Clerk (ES) - implemented by 'CREATE TABLE clerk'. Subsumption of User to Clerk entity sets implemented using the PRIMARY(ssn) from User table.

  NOTE: The subclasses of Person are done here to provide various roles and permissions to different entities in the Person table. It may be worth re-factoring into role and permissions tables in the future.
- Precinct (ES) - implemented by 'CREATE TABLE precinct'.
- Report (ES), reportsOn (R), files (R) - implemented by report_reportson_files.
- Crime (ES) - implemented by 'CREATE TABLE crime'.
- worksOn (ES) - implemented by 'CREATE TABLE worksOn'.
- Investigation (ER), relatesTo (R) - implemented by 'CREATE TABLE investigation_relatesto'
- assignedTo (R) - implemented by 'CREATE TABLE assignedTo'.
- Suspect (ES), isOn(R) - implemented by 'CREATE TABLE suspect_isOn'. This table also includes the hasA relationship to the Person entity set.
- StatusOption (ES) - implemented by 'CREATE TABLE statusoption'.
- SuspectStatus (ES), hasStatus (R), hasOption(R) - implemented by 'CREATE TABLE status_hasstatus_hasoption'.

Assertions and Constraints
Additional assertions can be found in the 'Trigger scripts.sql' file.

- Clerks and Standard Users are mutually exclusive - implemented by 'CREATE TRIGGER clerkNotStduser' and 'CREATE TRIGGER stduserNotClerk'.
- Status options must be at least four characters long - implemented by 'CREATE TRIGGER statusOptionName'.
- The Criminal Justice Report Application does not store police reports from before the 20th century - implemented by 'CREATE TRIGGER datelimit'.
- Detectives cannot be assigned to a case that they are a suspect in - implemented by 'CREATE TRIGGER userdetectiveNotSuspect'.
- Reports cannot be empty. Must be a minimum of five words - implemented by 'CREATE TRIGGER minReportLength'.
- Social security numbers of each person must be valid - implemented by 'CREATE TRIGGER ssnLength'.
- User passwords must contain a special character - implemented by 'CREATE TRIGGER userPassword'.

Functionality implemented by SQL queries

As was noticeable in the overview section, many queries model the role that a standard user would take. The standard user would perhaps be a researcher from a university wanting to access various crime statistics or a member of the public wanting to see which crimes occur near where they live. Many of the queries that we wrote reflect support for these standard user types.

1. Select the max number of investigations assigned to any detective.
   A query such a this would allow a standard user like a university researcher to find out which detectives are the most active out of all the precincts that are tracked by this database. The researcher might then use this information to research the personality traits of the most active detectives.

2. Select SSN's of all suspects who have status arrested.
   This query would most likely be used by a university researcher. The researcher might use this information from the database to find out how many suspects are currently being detained by the police department.

3. Select names of all officers who worked on a report in 2014.
   This query may be used by a member of the public that wants to see the list of officers who filed a report in a particular year. In this case 2014.

4. Select ranks of all officers who have worked on at least 10 reports, grouping by cName of report.
   This query would likely be very useful for a university researcher. The researcher may be interested in whether officers who have a higher rank are involved in more reports than officers with a lower rank. They may also be interested in the severity of the crimes which are investigated by officers of a higher rank.

5. Select name of precincts with at least 5 investigations associated with them.
   This query would most likely be of interest to a member of the public. That member of the public could be interested in how many ongoing investigations there are for crimes in the police precinct where they live.

6. Select reports where it shares both clerk and crime type with at least one other report.
   This particular query could be used by a researcher to find out which clerks in a police department file reports on the same crime often.

7. Multiply all reportNums by 10 (update query) or query to insert some tuple.
   This general query supports the functionality of our application by allowing the various employees of the police department to update the database with their respective reports and data.

8. Select iNumber of investigation(s) for which number of suspects being investigated for it is minimum.
   This query could be used by a university researcher to find the investigations that have the least suspects. The researcher could then use further queries to find out which crime types typically have the least suspects involved in the investigation.

9. Select names of all clerks who have never filed report for investigation with suspect under investigation.
   This query could perhaps be used by a member of the press (as a standard user). The member of the press might be interested in when it is in the legal process that a suspect is placed under an investigation.

10. Select name, status, date for each suspect order by date.
    This query may be used by a university researcher to see whether more suspects are apprehended on a particular date. They could also use the information of the name and status of the suspect to find out more about the suspects profile.

Building and populating the database

The database can be initialized via terminal with the following command:

`mysql -u <USERNAME> -p < "batch.sql"`

This will delete the policedatabase (if it exists), and then create the tables, create the assertions, and import the initial mock data.

References

- https://stackoverflow.com/questions/2981930/mysql-trigger-to-prevent-insert-under-certain-conditions
- http://mysqlserverteam.com/new-and-old-ways-to-emulate-check-constraints-domain/