CS 157A Project 2

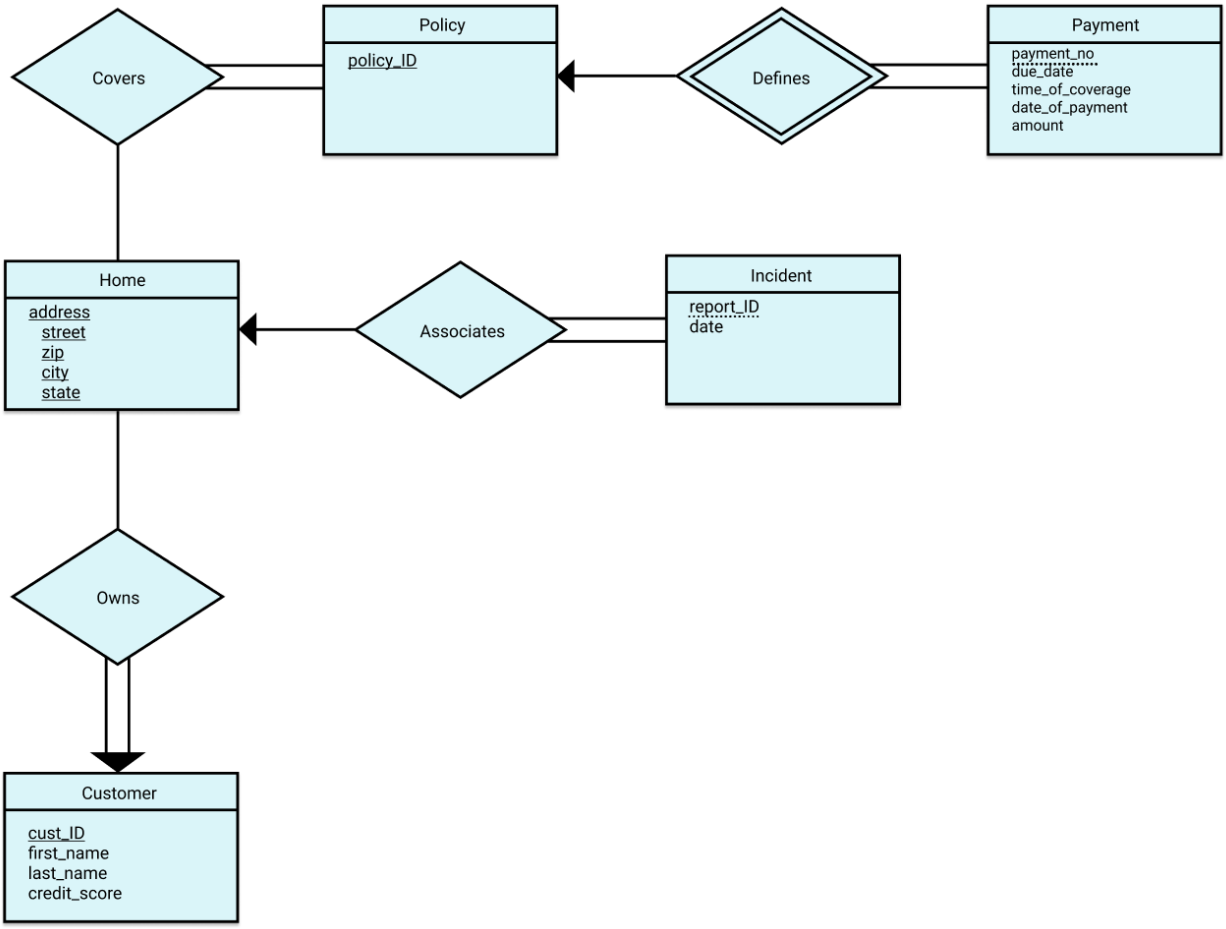
May 14, 2021

Andrew Shinjo
Camille Barker
Sergio Gutierrez
Shanon Gao
Shayanna Gatchalian

Introduction

The goal of this project is to address the problem statement which claims we are a consulting company. We have been asked to develop a relational database design for a company that sells home insurance. We are then given some information about the relationship between the customer, home, incident, policy, payment, and how these entities relate to one another. We accomplish this by creating an ER diagram that illustrates this interwoven relationship in a coherent way in which stakeholders who don't have technical knowledge can understand.

ER Model



Our ER model has five entities (shown as rectangles) based on the information the insurance company has given us: customer, home, incident, policy, and payment. Each entity has attributes that will later define information about each unique object of that

entity. Four relationships (shown as diamonds) connect each entity to another entity, with different edges and arrows signifying different cardinalities between different entities.

- Every customer owns 1 or more homes.
- Every incident has 1 home associated with it.
- Every policy covers 1 or more homes.
- Every policy defines 1 or more payments.

With this ER diagram, we can then derive the structure for the relational schema.

Relational Schema Derived from the ER Model

In the previous section, we described the database design using an E-R model. After satisfied with our E-R model for a home insurance company we can move on to deriving the relational schema from the ER model. Our database model has 5 entities with their relative attributes:
Customer(cust_id, first_name, last_name, credit_score)
Home(cust_id, policy_id, street, zip, city, state)
Incident(report_ID, street, zip, city, state, date)
Policy(policy_ID)
Payment(payment_no, policy_ID, due_date, time_of_coverage, date_of_payment, amount)

In the relational schema, we created each of these entities as a table. Furthermore, we chose to make all attributes `not null` attributes to ensure that no column can have a null value. In the `Home` table, its primary key is `street`, `zip`, `city`, and state and it has two foreign keys, `cust_id` which references `Customer`, and `policy_id` which references `Policy`. In the `Customer` table, its primary key is `cust_id` and there are no foreign keys. In the `Incident` table, its primary key is `report_ID` and it has foreign keys `street`, `zip`, `city`, and `state` which references `Home`. In the `Policy` table, its primary key is `policy_ID`. In the `Payment` table, its primary key is `payment_no` and `policy_ID`. `Payment` also has one foreign key, `policy_ID` which references `Policy`. We do not have any more constraints added to the `not null` constraint placed on all attributes.

Schema in SQL:

```
create table Home(
      cust_id           integer not null,
      policy_id         integer not null,
      street            varchar(64) not null,
      zip               integer not null,
      city              varchar(65) not null,
      state             varchar(65) not null,
      primary key (street, zip, city, state),
      foreign key (cust_id) references Customer,
      foreign key (policy_id) references Policy
```

```
      );

create table Customer(
      cust_id           integer not null primary key,
      first_name        varchar(64) not null,
        last_name       varchar(64) not null,
        credit_score    integer not null
      );

create table Incident(
      report_ID         integer not null primary key,
      street            varchar(64) not null,
      zip               integer not null,
      city              varchar(65) not null,
      state             varchar(65) not null,
      date              varchar(65) not null,
      foreign key (street, zip, city, state) references Home
      );

create table Policy(
      policy_ID   integer not null primary key
      );

create table Payment(
      payment_no        integer not null,
      policy_ID         integer not null,
      due_date          varchar(10) not null,
      time_of_coverage  integer not null,
      date_of_payment   varchar(10) not null,
      amount            integer not null,
      primary key (payment_no, policy_ID),
      foreign key (policy_ID) references Policy
      );
```

Normalized Relational Schema

Home has a primary key that consists of the attributes Street, Zip, City, and State. The four attributes can be used to uniquely identify a cust_id. Therefore Street, Zip, City, State→cust_id is a functional dependency. We know that Street, Zip, City, State is a primary key so it is also a superkey. Therefore, Home is in BCNF form.

Customer's primary key is cust_id. Three functional dependencies that exist are cust_id → first_name, cust_id → last_name, and cust_id → credit_score. Since cust_id is the primary key, it is also a superkey. Therefore, Customer is in BCNF form.

Incident's primary key is report_ID. report_ID has a functional dependency with every other attribute in the schema because a unique report_ID should always correspond to the same street, zip code, city, state, and date. Since report_ID is the primary key, we know that it is also the super key. Therefore, Incident is in BCNF form.

Policy has no functional dependencies so it is in BCNF form.

Payment's primary key is payment_no. Four functional dependencies that exist are payment_no→due_date, payment_no → time_of_coverage, payment_no → date_of_payment, and payment_no → amount. Since payment_no is the primary key, we know that it is also a super key. Therefore, Payment is in BCNF form.

Sample Data and SQL Queries

As a company that provides insurance policies for homes, organizing our customer's information is of extreme importance. The following are our 5 example queries, giving results of typical questions either the insurance company or the customer would want to know. The subsequent section shows how our database would organize our clients' data and their insurance policies with sample data derived from each of the team members.

Queries on the sample data:

1. How many incidents does each home have?

```
select street, city, zip, state, count(report_ID) as num_incidents from
Home left natural join Incident group by street;
```

```
sqlite> select street, city, zip, state, count(report_ID) as num_incidents from Home left natural join Incident group by
 street;
street                  city          zip    state  num_incidents
----------------------  ------------  -----  -----  -------------
1000 Apple Street       San Jose      95125  CA     5
1020 Fairburn Cambelton Rd  Fairburn      67384  GA     1
124 Conch St            Bikini Bottom 34526  HI     2
1890 Banana Street      San Jose      95123  CA     0
20 Ingram St            Forest Hills  11375  NY     1
3000 Bernardo Ave       Sunnyvale     94087  CA     0
4545 Dover Rd           Redwood City  94063  CA     0
600 Marsh Rd            Menlo Park    94025  CA     3
6666 Hell St            Somewhere     66666  FL     1
777 Heaven St           Sky           77777  HI     2
844 Pennsylvania Ave    Brooklyn      34859  NY     1
```

2. In total, how much money has a customer paid for their policy?

```
select cust_id, first_name, last_name, sum (amount) from Customer Natural
join Home natural join Policy natural join Payment group by policy_ID
having cust_id=[insert cust_id here];

(Used cust_id 89050 in the example below)
```

```
sqlite> select cust_id, first_name, last_name, sum (amount) from Customer
   ...> natural join Home natural join Policy natural join Payment group by policy_ID having cust_id=89050;
89050|Andrew|Shinjo|17200
```

3. Which customer(s) whose address does not contain the letter "a" anywhere in the

customer's street, city, and state name?

```
Select first_name, last_name from customer natural join home where street
NOT like"%a%" and city NOT like "%a%" and state NOT like "%a%";
```

```
sqlite> Select first_name, last_name from customer natural join home where street NOT like"%a%" and city NOT like "%a%"
and state NOT like "%a%";
Spongebob|Squarepants
The|Devil
```

4. Show the names and credit scores of customers with a credit score of over 700. Order in descending order.

```
select first_name, last_name, credit_score from Customer where
credit_score > 700 order by (credit_score) desc;
```

```
sqlite> select first_name, last_name, credit_score from Customer where credit_score > 700 order by (credit_score) desc;
first_name  last_name    credit_score
----------  -----------  ------------
Steve       Rodgers      920
Tony        Stark        850
Camille     Barker       800
The         Angel        777
Dustin      Barker       770
Spongebob   Squarepants  720
sqlite>
```

5. How long has each home been covered?

```
select street, zip, city, state, sum(time_of_coverage) from Home natural
join Payment group by street;
```

```
sqlite> select street, zip, city, state, sum(time_of_coverage) from Home natural join Payment group by street;
street                    zip    city          state  sum(time_of_coverage)
------------------------  -----  ------------  -----  --------------------
1000 Apple Street         95125  San Jose      CA     24
1020 Fairburn Cambelton Rd  67384  Fairburn      GA     30
124 Conch St              34526  Bikini Bottom  HI     10
1890 Banana Street        95123  San Jose      CA     30
20 Ingram St              11375  Forest Hills  NY     27
3000 Bernardo Ave         94087  Sunnyvale     CA     24
4545 Dover Rd             94063  Redwood City  CA     24
600 Marsh Rd              94025  Menlo Park    CA     24
6666 Hell St              66666  Somewhere     FL     198
777 Heaven St             77777  Sky           HI     14
844 Pennsylvania Ave      34859  Brooklyn      NY     24
sqlite>
```

Sample data written in SQL:

```
insert into Customer values (358262, 'Camille', 'Barker', 800);
insert into Home values (358262, 299622, '4545 Dover Rd', 94063, 'Redwood City',
'CA');
insert into Home values (358262, 299622, '3000 Bernardo Ave', 94087,
'Sunnyvale', 'CA');
insert into Policy values (299622);
insert into Payment values (047187, 299622, '01/05/2020', '12', '01/03/2020',
45000);
insert into Payment values (047190, 299622, '01/05/2021', '12', '12/22/2020',
45000);
insert into Customer values (931850, 'Dustin', 'Barker', '770');
insert into Home values (931850, 960322, '600 Marsh Rd', 94025, 'Menlo Park',
'CA');
insert into Policy values (960322);
```

```
insert into Payment values (047188, 960322, '01/05/2020', '12', '01/07/2020',
20000);
insert into Payment values (047191, 960322, '01/05/2021', '12', '01/05/2021',
20000);
insert into Incident values (353984, '600 Marsh Rd', 94025, 'Menlo Park', 'CA',
'10/31/2000');
insert into Incident values (353986, '600 Marsh Rd', 94025, 'Menlo Park', 'CA',
'01/01/2021');
insert into Incident values (353990, '600 Marsh Rd', 94025, 'Menlo Park', 'CA',
'05/05/2021');
insert into Customer values (000001, 'Tony', 'Stark', 850);
insert into Home values (000001, 098765, '1020 Fairburn Cambelton Rd', 67384,
'Fairburn', 'GA');
insert into Incident values (172638, '1020 Fairburn Cambelton Rd', 67384,
'Fairburn', 'GA', '01/10/2016');
insert into Policy values (098765);
insert into Payment values (000546, 098765, '05/29/2015', 10, '05/22/2015',
20000);
insert into Payment values (000547, 098765, '09/26/2015', 10, '09/24/2015',
20000);
insert into Payment values (000548, 098765, '12/30/2015', 10, '12/24/2015',
20000);
insert into Customer values (000808, 'Spongebob', 'Squarepants', 720);
insert into Home values (000808, 567567, '124 Conch St', 34526, 'Bikini Bottom',
'HI');
insert into Incident values (192658, '124 Conch St', 34526, 'Bikini Bottom',
'HI', '04/07/2010');
insert into Incident values (192659, '124 Conch St', 34526, 'Bikini Bottom',
'HI', '08/18/2010');
insert into Policy values (567567);
insert into Payment values (000653, 567567, '07/13/2010', 5, '07/15/2010',
1400);
insert into Payment values (000654, 567567, '11/15/2010', 5, '11/10/2010',
1400);
insert into Customer values(89050, 'Andrew', 'Shinjo', 432);
insert into Policy values(23);
insert into Home values(89050, 23, '1000 Apple Street', 95125, 'San Jose',
'CA');
insert into Incident values(294324, '1000 Apple Street', 95125, 'San Jose',
'CA', '01/01/2012');
insert into Incident values(294325, '1000 Apple Street', 95125, 'San Jose',
'CA', '01/02/2012');
insert into Incident values(294326, '1000 Apple Street', 95125, 'San Jose',
'CA', '01/03/2012');
insert into Incident values(294327, '1000 Apple Street', 95125, 'San Jose',
'CA', '01/04/2012');
```

```
insert into Incident values(294328, '1000 Apple Street', 95125, 'San Jose',
'CA', '01/05/2012');
insert into Payment values(153, 23, '01/30/2012', 12, '01/30/2012', 7200);
insert into Payment values(154, 23, '02/30/2012', 12, '02/29/2012', 10000);
insert into Customer values(17032, 'Bobby', 'Bob', 432);
insert into Policy values(37);
insert into Home values(17032, 37, '1890 Banana Street', 95123, 'San Jose',
'CA');
insert into Payment values(216, 37, '05/12/2019', 30, '05/01/2019', 9600);
insert into Customer values (040561, 'Steve', 'Rodgers', 920);
insert into Home values (040561, 562761, '844 Pennsylvania Ave', 34859,
'Brooklyn', 'NY');
insert into Incident values (489652, '844 Pennsylvania Ave', 34859, 'Brooklyn',
'NY', '05/10/2017');
insert into Policy values (562761);
insert into Payment values (000600, 562761, '04/28/2017', 8, '04/21/2017',
30000);
insert into Payment values (000601, 562761, '08/30/2017', 8, '08/23/2017',
30000);
insert into Payment values (000602, 562761, '11/27/2017', 8, '11/20/2017',
30000);
insert into Customer values (934032, 'Peter', 'Parker', 493);
insert into Home values (934032, 047695, '20 Ingram St', 11375, 'Forest Hills',
'NY');
insert into Incident values (357839, '20 Ingram St', 11375, 'Forest Hills',
'NY', '05/10/2017');
insert into Policy values (047695);
insert into Payment values (000324, 047695, '03/17/2018', 9, '03/10/2018',
20000);
insert into Payment values (000325, 047695, '06/20/2018', 9, '06/12/2018',
20000);
insert into Payment values (000326, 047695, '08/22/2018', 9, '08/15/2018',
20000);
insert into Customer values (000666, 'The', 'Devil', 666);
insert into Home values (000666, 666666, '6666 Hell St', 66666, 'Somewhere',
'FL');
insert into Incident values (777666, '6666 Hell St', 66666, 'Somewhere', 'FL',
'01/01/1950');
insert into Policy values (666666);
insert into Payment values (000660, 666666, '03/23/2004', 66, '03/18/2004',
60000);
insert into Payment values (000661, 666666, '08/22/2003', 66, '08/20/2003',
60000);
insert into Payment values (000662, 666666, '02/05/2002', 66, '02/02/2002',
60000);
insert into Customer values (000777, 'The', 'Angel', 777);
insert into Home values (000777, 777777, '777 Heaven St', 77777, 'Sky', 'HI');
```

```
insert into Incident values (666777, '777 Heaven St', 77777, 'Sky', 'HI',
'012/25/2000');
insert into Incident values (666778, '777 Heaven St', 77777, 'Sky', 'HI',
'012/20/2000');
insert into Policy values (777777);
insert into Payment values (000770, 777777, '06/09/2001', 7, '06/06/2001',
1700);
insert into Payment values (000771, 777777, '09/18/2009', 7, '09/11/2009',
1700);
```

Conclusion

We were given a problem statement that had us create an ER diagram to model the relationship between Home, Incident, Policy, Customer and Payment. We worked on the ER diagram together through the web application “figma”. In this application we were able to create the visual diagram presented. We also collaborated through Zoom via voice chat and a shared document via Google documents. Based on our ER diagram we were able to create relational schemas derived from the model. For normalization of the relational schema we found that all five entities were in BCNF form therefore they have gone through the steps of normalization. We each created two customer entities in order to populate our sample data. Lastly, we each made up one unique SQL query (total of 5) based on the data we created and tested to see if they were correct with screenshots.