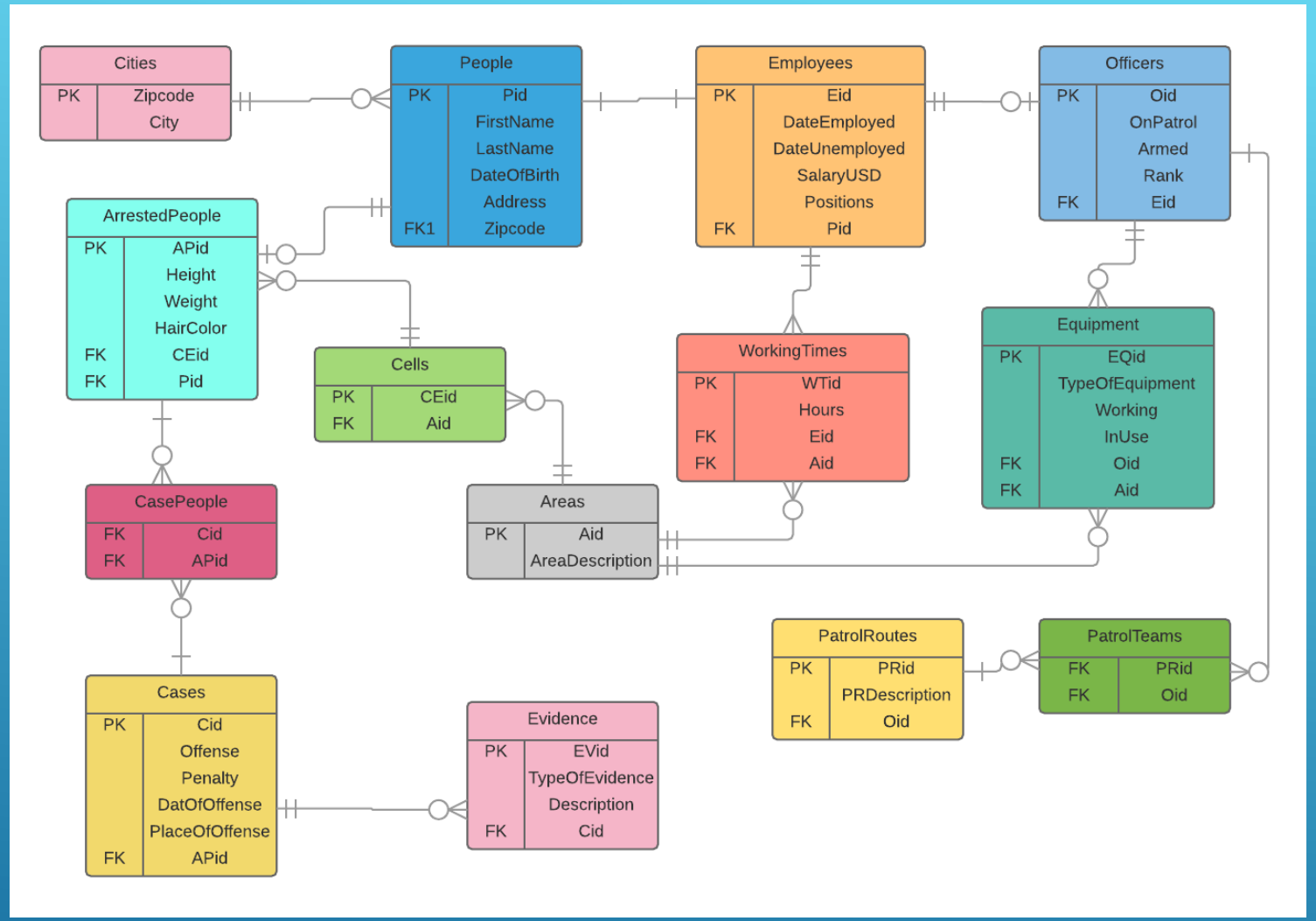
CODD COUNTY POLICE DEPARTMENT

By Ryan Rendeiro

TABLE OF CONTENTS

EXECUTIVE SUMMARY

- The database for the codd county police department is outlined and describe with in this documents. The purpose of this database is to keep track of employees, police officers, arrested suspects, and items and equipment related to their activities. Such related activities and items as equipment, evidence, and when they work. This database also catalogs where police officers work with in the police department premises and where they operate outside of the premises. This will empower administration of the police department to be able to analyze and extract useful and informative data about the police department. Ultimately the database should be able to provide useful control over the departments data while being normalized and elegantly serving the department's needs.



ENTITY RELATION DIAGRAM

PEOPLE

- Lists all people with in the system with attributes and functional dependencies.
- CREATE TABLE people (
- Pid text NOT NULL UNIQUE,
- FirstName text NOT NULL,
- LastName text NOT NULL,
- DateOfBirth text NOT NULL,
- Address text NOT NULL,
- Zipcode text NOT NULL,
- FOREIGN KEY (Zipcode) REFERENCES cities(Zipcode),
- PRIMARY KEY (Pid,Zipcode)
-);
- Functional Dependencies Pid → FirstName, LastName, DateOfBirth, Address, Zipcode
- Sample data on the next page

	pid text	firstname text	lastname text	dateofbirth text	address text	zipcode text
2	A002	Bill	Bill	04/20/1986	13 Awesome Rd	86754
3	A003	Have	Bill	04/20/1986	14 Awesome Rd	86754
4	A004	An	Bill	09/12/1983	15 Awesome Rd	86754
5	A005	Awesome	Bill	09/12/1983	16 Awesome Rd	86754
6	A006	Hat	Bill	09/12/1983	17 Awesome Rd	86754
7	A007	Pablo	Rivas	06/15/1984	69 Database Blvd	95854
8	A008	Henry	Alopiza	10/02/1976	2 Little Wood Rd	95855
9	A009	Magarita	Piza	01/29/1988	79 Main St	95854
10	A010	Sal	Diracki	08/17/1968	1 Enigo Ave	86754
11	A011	Inigo	Montoya	03/06/1977	34 Mellamoes Rd	95855
12	A012	Lily	Vasdina	04/13/1966	11 Awesome Rd	86754
13	A013	Jefferey	O Doyle	03/18/1991	103 Database Blvd	95854
14	A0014	Lee	Choo	12/24/1983	25 Codd St	95855
15	A015	Michael	Wittensberg	08/01/1985	28 Balabras Rd	86754
16	A016	Magic	Johnson	03/31/1975	20 Awesome Rd	86754
17	A017	Jackie	Baliz	05/21/1992	22 Main St	95854
18	A018	Alace	Johnson	10/31/1981	22 Awesome Rd	86754
19	A019	Johnny	Jones	02/29/1986	28 Codd St	95855
20	A020	Jimmy	Crowley	09/21/1975	2 Mellamoes Rd	95855
21	A021	Sarah	Goodman	11/26/1988	3 Enigo Ave	86754
22	A022	Alejandro	Menedez	07/19/1988	72 Database Blvd	95855
23	A023	Vladamir	Rusikov	11/01/1979	46 Balabras Rd	86754
24	A024	John	Cena	04/03/1977	1 Mellamoes Rd	95855

CITIES

- Lists all cities with in the system with their attributes and functional dependencies
- CREATE TABLE cities (
- Zipcode text NOT NULL UNIQUE,
- City text NOT NULL,
- PRIMARY KEY (Zipcode)
-);
- Functional Dependencies Zipcode→ City

	zipcode text	city text
1	86754	Coddington Township
2	95855	Data Ridge
3	95854	Selecton

EMPLOYEES

- Lists all employees that exist with in the system with their attributes and functional dependencies
- CREATE TABLE Employees (
- Eid text NOT NULL UNIQUE,
- DateEmployed text NOT NULL,
- DateUnemployed text,
- SalaryUSD text NOT NULL,
- Positions text NOT NULL,
- Pid text NOT NULL,
- FOREIGN KEY (Pid) REFERENCES people (Pid),
- PRIMARY KEY (Eid,Pid)
-);
- Functional Dependencies Eid → DateEmployed, DateUnemployed, SalaryUSD, Positions, Pid
- Sample data on next page

	eid text	dateemployed text	dateunemployed text	salaryusd integer	positions text	pid text
1	E001	02/21/2005		28000	Janitor	A020
2	E002	03/13/2005		28000	Janitor	A001
3	E003	08/24/2005		50000	Police Officer	A021
4	E004	12/21/2004		50000	Police Officer	A002
5	E005	01/01/2006		50000	Police Officer	A009
6	E006	06/15/2005		52000	Police Officer	A011
7	E007	07/18/1984		62000	Police Officer	A012
8	E008	07/10/1987		69000	Police Officer	A010
9	E009	02/21/2001		54000	Police Officer	A017
10	E010	02/21/2003		50000	Police Officer	A004
11	E011	02/21/2002		57000	Police Officer	A003
12	E012	02/21/1994		59000	Police Officer	A013
13	E013	02/21/2005		54000	Police Officer	A018
14	E014	12/21/2005		54000	Police Officer	A016
15	E015	11/09/2004		52000	Police Officer	A015
16	E016	01/17/2007		50000	Police Officer	A008
17	E017	05/04/2003		42000	Mechanic	A006

OFFICERS

- Lists all police officers in the system with their attributes and functional dependencies
- CREATE TABLE Officers (
- Oid text NOT NULL UNIQUE,
- OnPatrol text NOT NULL,
- Armed text NOT NULL,
- Rank text NOT NULL,
- Eid text NOT NULL,
- FOREIGN KEY (Eid) REFERENCES Employees (Eid),
- PRIMARY KEY (Oid,Eid)
-);
- Functional Dependencies Oid→ OnPatrol, Armed, Rank, Eid

	oid text	onpatrol text	armed text	rank text	eid text
1	001	Yes	Yes	Patrol Officer	E003
2	002	Yes	Yes	Patrol Officer	E004
3	003	Yes	Yes	Patrol Officer	E005
4	004	No	Yes	Corporal	E006
5	005	No	Yes	Lieutenant	E007
6	006	No	Yes	Chief	E008
7	007	No	Yes	Sergeant	E009
8	008	Yes	Yes	Patrol Officer	E010
9	009	No	Yes	Senior Sergeant	E011
10	010	No	Yes	Senior Sergeant	E012
11	011	No	Yes	Sergeant	E013
12	012	No	Yes	Sergeant	E014
13	013	No	Yes	Corporal	E015
14	014	Yes	Yes	Patrol Officer	E016

WORKING TIME

- Lists all the times employees work and the attributes about those times and their functional dependencies
- CREATE TABLE WorkingTimes (
- WTid text NOT NULL UNIQUE,
- Hours text NOT NULL,
- Eid text NOT NULL,
- Aid text NOT NULL,
- FOREIGN KEY (Eid) REFERENCES Employees (Eid),
- FOREIGN KEY (Aid) REFERENCES Areas (Aid),
- PRIMARY KEY (WTid,Eid,Aid)
-);
- Functional Dependencies WTid→ Hours, Eid, Aid

	wtid text	hours text	eid text	aid text
1	WT01	8AM to 8PM	E001	AR01
2	WT02	8AM to 8PM	E002	AR03
3	WT03	8AM to 8PM	E006	AR01
4	WT04	8AM to 8PM	E015	AR01
5	WT05	8AM to 8PM	E013	AR01
6	WT06	8AM to 8PM	E014	AR01
7	WT07	8AM to 8PM	E007	AR01
8	WT08	8AM to 8PM	E008	AR01
9	WT09	8AM to 8PM	E009	AR03
10	WT10	8AM to 8PM	E011	AR04
11	WT11	8AM to 8PM	E012	AR06
12	WT12	8AM to 8PM	E016	AR07
13	WT13	8AM to 8PM	E010	AR07
14	WT14	8AM to 8PM	E003	AR07
15	WT15	8PM to 8AM	E004	AR07
16	WT16	8PM to 8AM	E005	AR07
17	WT17	8AM to 8PM	E017	AR05

EQUIPMENT

- Lists all attributes and data on equipment in the system with thei functional dependencies
- CREATE TABLE Equipment (
- EQid text NOT NULL UNIQUE,
- TypeOfEquipment text NOT NULL,
- Working text NOT NULL,
- InUse text NOT NULL,
- Oid text NOT NULL,
- Aid text NOT NULL,
- FOREIGN KEY (Oid) REFERENCES Officers (Oid),
- FOREIGN KEY (Aid) REFERENCES Areas (Aid),
- PRIMARY KEY (EQid,Oid,Aid)
-);
- Functional Dependencies EQid→ TypeOfEquipment, Working, InUse, Oid, Aid

	eqid text	typeofequipment text	working text	inuse text	oid text	aid text
1	EQ01	Gun	Yes	Yes	O01	AR04
2	EQ02	Gun	Yes	Yes	O02	AR04
3	EQ03	Gun	Yes	Yes	O03	AR04
4	EQ04	Gun	Yes	No	O08	AR04
5	EQ05	Gun	No	No	O08	AR04
6	EQ06	Car	Yes	Yes	O01	AR06
7	EQ07	Car	No	No	O08	AR06
8	EQ08	Kevlar Vest	Yes	Yes	O01	AR02
9	EQ09	Kevlar Vest	Yes	Yes	O02	AR02
10	EQ10	Kevlar Vest	Yes	No	O03	AR02
11	EQ11	Kevlar Vest	No	No	O08	AR02

PATROL ROUTES

- Lists all attributes and functional dependencies about patrol routes
- CREATE TABLE PatrolRoutes (
- PRid text NOT NULL UNIQUE,
- PRDescription text NOT NULL,
- PRIMARY KEY (PRid)
-);Functional Dependencies PRid→ PRDescription,

	prid text	prdescription text
1	PR01	Coddington Township Awesome Rd to Enigo Ave
2	PR02	Coddington Township Awesome Rd to Balabras Rd
3	PR03	Data Ridge Mellamoes Rd to Little Wood Rd
4	PR04	Data Ridge Litle Wood to Codd St
5	PR05	Selecton Database Blvd to Main St

AREAS

- Lists all areas with their attributes and functional depedncies
- CREATE TABLE Areas (
- Aid text NOT NULL UNIQUE,
- AreaDescription text NOT NULL,
- PRIMARY KEY (Aid)
-);
- Functional Dependencies Aid→ AreaDescription

	aid text	areadescription text
1	AR01	Offices
2	AR02	Locker Room
3	AR03	Holding Cells
4	AR04	Armory
5	AR05	Parking Lot
6	AR06	Storage Room
7	AR07	Off the Premises

ARRESTED PEOPLE

- Lists all arrested people with their attributes and their functional dependencies
- CREATE TABLE Arrestedpeople (
- APid text NOT NULL UNIQUE,
- Height text NOT NULL,
- Weight text NOT NULL,
- HairColor text NOT NULL,
- CEid text NOT NULL,
- Pid text NOT NULL,
- FOREIGN KEY (CEid) REFERENCES cells (CEid),
- FOREIGN KEY (Pid) REFERENCES people (Pid),
- PRIMARY KEY (APid,Pid,CEid)
-);
- Functional Dependencies APid→ Height, Weight, HairColor, Ceid, Pid

	apid text	height text	weight text	haircolor text	ceid text	pid text
1	AP01	5ft 7in	168lbs	Brown	CE02	A005
2	AP02	6ft 0in	201lbs	Black	CE01	A007
3	AP03	4ft 11in	147lbs	Brown	CE02	A014
4	AP04	5ft 3in	159lbs	Red	CE02	A019
5	AP05	5ft 9in	185lbs	Brown	CE02	A022
6	AP06	6ft 3in	227lbs	Blonde	CE02	A023
7	AP07	6ft 1in	251lbs	Brown	CE03	A024

CELLS

- Lists all cells within the system and their attributes and functional dependencies
- CREATE TABLE cells (
- CEid text NOT NULL UNIQUE,
- Aid text NOT NULL,
- FOREIGN KEY (Aid) REFERENCES Areas (Aid),
- PRIMARY KEY (CEid, Aid)
-);
- Functional Dependencies CEid→ Aid

	ceid text	aid text
1	CE01	AR03
2	CE02	AR03
3	CE03	AR03

CASES

- Lists all cases within the system with their attributes and functional dependencies
- CREATE TABLE Cases (
- Cid text NOT NULL UNIQUE,
- Offense text NOT NULL,
- Penalty text NOT NULL,
- DateOfOffense text NOT NULL,
- PlaceOfOffense text NOT NULL,
- PRIMARY KEY (Cid)
-); Functional Dependencies Cid→ Offense, Penalty, DateOfOffense, PlaceOfOffense
- Sample data on next page

	cid text	offense text	penalty text	dateofoffense text	placeofoffense text
1	C01	Arson	Imprisonment	03/19/2013	Coddington Township
2	C02	Disturbing the Peace	Fined	4/20/2015	Data Ridge
3	C03	Grand Theft Auto	Imprisonment	2/11/2014	Selecton
4	C04	1st Degree Murder	Imprisonment	03/19/2013	Coddington Township
5	C05	Vandalism	Fined	11/22/2012	Selecton
6	C06	Tax Evasion	Imprisonment	08/15/2011	Data Ridge

EVIDENCE

- Lists all pieces of evidence within the system with their attributes and functional dependencies
- CREATE TABLE Evidence (
- EVid text NOT NULL UNIQUE,
- TypeOfEvidence text NOT NULL,
- Description text NOT NULL,
- Cid text NOT NULL,
- FOREIGN KEY (Cid) REFERENCES Cases (Cid),
- PRIMARY KEY (EVid, Cid)
-);
- Functional Dependencies EVid→ TypeOfEvidence, Description, Cid
- Sample data on next page

	evid text	typeofevidence text	description text	cid text
1	EV01	Eye Witness	Suspect was making bad puns about cods	C02
2	EV02	Video Tape	Shows suspects Setting fire to building	C01
3	EV03	Eye Witness	First Responders found burnt bodies in the burned out building	C04
4	EV04	Documents	Documents of an audit of the suspect showing that they commited tax evasion	C06
5	EV05	Picture	Picture of vadalism with suspects name signed next to it	C05

PATROL TEAMS

- Lists all patrol teams and their attributes within the database and their functional dependencies
- CREATE TABLE PatrolTeams (
- Oid text NOT NULL,
- PRid text NOT NULL,
- FOREIGN KEY (Oid) REFERENCES Officers (Oid),
- FOREIGN KEY (PRid) REFERENCES PatrolRoutes(PRid),
- PRIMARY KEY (Oid, PRid)
-);
- Functional Dependencies PRid→Oid

	oid text	prid text
1	001	PR01
2	008	PR01
3	001	PR02
4	008	PR02
5	002	PR03
6	003	PR03
7	002	PR04
8	003	PR04
9	014	PR05

CASE PEOPLE

- Lists all case people and their attributes and functional dependencies
- CREATE TABLE CasePeople (
- Cid text NOT NULL,
- APid text NOT NULL,
- FOREIGN KEY (Cid) REFERENCES Cases (Cid),
- FOREIGN KEY (APid) REFERENCES ArresedPeople (APid),
- PRIMARY KEY (Cid, APid)
-);
- Functional Dependencies Cid→APid

	cid text	apid text
1	C01	AP01
2	C01	AP03
3	C01	AP04
4	C02	AP02
5	C03	AP05
6	C04	AP01
7	C04	AP03
8	C04	AP04
9	C05	AP07
10	C06	AP06

VIEW ARRESTED SUSPECTS

This view creates a table that puts the names of arrested people with their cell numbers making it easy keep track of arrested people

```
Create View ArrestedSuspects AS
  Select FirstName, LastName, ArrestedPeople.Ceid as Cell Number
  From people
  Inner join ArrestedPeople
  ON people.pid = ArrestedPeople.pid
  Inner join Cells
  ON Cells.Ceid = ArrestedPeople.Ceid;
```

	firstname text	lastname text	cellnumber text
1	Awesome	Bill	CE02
2	Pablo	Rivas	CE01
3	Lee	Choo	CE02
4	Johnny	Jones	CE02
5	Alejandro	Menedez	CE02
6	Vladamir	Rusikov	CE02
7	John	Cena	CE03

This view will create a table that will show which police officer broke which piece of equipment allowing the user to determine both what needs to be repaired and if the offending officer should get new equipment

```
Create View WhoBrokeIt as
  Select FirstName, LastName, Officer.Oid as OfficerID, Equipment.Eqid as EquipmentID, Equipment.TypeOfEquipment as EquipmentType
  From People
  Inner Join Employees
  On people.pid = employees.pid
  inner join Officer
  On employees.eid = Officer.eid
  inner join equipment
  On officer.Oid = Equipment.oid
  Order by Officer.Oid;
```

	firstname text	lastname text	officerid text	equipmentid text	equipmenttype text
1	An	Bill	008	EQ11	Kevlar Vest
2	An	Bill	008	EQ07	Car
3	An	Bill	008	EQ05	Gun

VIEW WHO BROKE IT

This view create a table that shows the user every police officers name, id and patrol route, for police officers who have patrol routes, allowing the user to keep track of on duty officers that aren't at the police department.

```
Create View WhoPatrolsWhat as
  Select FirstName, LastName, Officer.Oid as OfficerID,
PatrolRoutes.PRDescription as PatrolRoute
  from people
  inner join employees
  ON people.pid = employees.pid
  inner join officers
  ON employees.eid = officers.eid
  inner join PatrolRoutes
  ON officers.Oid = PatrolRoutes.Oid
  Order by Officers.Oid;
```

	firstname text	lastname text	officerid text	patrolroute text
1	Sarah	Goodman	001	Coddington Township Awesome Rd to Balabras Rd
2	Sarah	Goodman	001	Coddington Township Awesome Rd to Enigo Ave
3	Bill	Bill	002	Data Ridge Mellamoes Rd to Little Wood Rd
4	Bill	Bill	002	Data Ridge Litle Wood to Codd St
5	Magarita	Piza	003	Data Ridge Mellamoes Rd to Little Wood Rd
6	Magarita	Piza	003	Data Ridge Litle Wood to Codd St
7	An	Bill	008	Coddington Township Awesome Rd to Enigo Ave
8	An	Bill	008	Coddington Township Awesome Rd to Balabras Rd
9	Henry	Alopiza	014	Selecton Database Blvd to Main St

VIEW WHO PATROLS WHAT

REPORTS

- The following query counts the amounts of police officers then employees and divides them to get the percentage of employees who are police officers.
- Select TRUNC (
- CAST((SELECT COUNT(Eid) as PoliceCount
- FROM Employees
- WHERE positions = 'Police Officer') as DECIMAL(5,2)
-) /
- (SELECT COUNT(Eid) as EmployeeCount
- FROM Employees) * 100) as PercentPolice

	percentpolice numeric
1	82

This query will get the user a percentage amount of how many people live in the town of Data Ridge.

```
Select TRUNC (
      CAST((SELECT COUNT(Pid) as WherePeopleLive
           FROM People
           INNER JOIN Cities ON
           People.Zipcode = Cities.Zipcode
           WHERE city = 'Data Ridge') as DECIMAL(5,2)
          ) /
          (SELECT COUNT(Pid) as People
           FROM People) * 100) as DataRidgePeople
```

	dataridgepeople numeric
1	29

REPORTS

This query will give the user a percentage of how many people live in coddington township.

```
Select TRUNC (
   CAST((SELECT COUNT(Pid) as WherePeopleLive
            FROM People
            INNER JOIN Cities ON
            People.Zipcode = Cities.Zipcode
            WHERE city = 'Coddington Township') as DECIMAL(5,2)
          ) /
          (SELECT COUNT(Pid) as People
           FROM People) * 100) as CoddingtonPeople
```

	coddingtonpeople numeric
1	54

REPORTS

REPORTS

- This query gives the user a percentage amount of how many people live in the town of Selecton
- Select TRUNC (
- CAST((SELECT COUNT(Pid) as WherePeopleLive
- FROM People
- INNER JOIN Cities ON
- People.Zipcode = Cities.Zipcode
- WHERE city = 'Selecton') as DECIMAL(5,2)
-) /
- (SELECT COUNT(Pid) as People
- FROM People) * 100) as SelectonPeople

	selectonpeople numeric
1	16

STORED PROCEDURE

This stored procedure gets called by the trigger that waits for new police officers then assigns then working times in the working times relation.

```
Create Or Replace Function AddWorkHours(refcursor) Returns refcursor AS $$
Declare
 p refcursor = $1;
Begin
        IF NEW.Positions = 'Police Officer' THEN
                Insert Into WorkingTimes (Eid) VALUES (NEW.Eid);
        END IF;
        RETURN p;
END;
$$
Language plpgsql;
```

TRIGGER

This Trigger waits for a new police officers to be added to the employees relation and then assigns them a spot in the working times table to ensure that they have times to work in the police department.

```
Create Trigger AddPoliceOfficer

After Insert on Employees

For Each Row

Execute Procedure AddWorkHours();
```

SECURITY

- The following roles would be used by authorized personnel to get access to certain parts of the database to update, insert, or lookup data and to give/revoke privileges as necessary.
- Admin Role

  Create Role Admin;

  Grant all on all tables

  In schema public

  To Admin;

Employees Role

  Create Role Employees;

  Grants Select On People, Employees, Officers, ArrestedPeople, Cells, Equipment, PatrolRoutes, WorkingTimes, Areas, Cases, and Evidence

  To Employees;

  Grants Insert On ArrestedPeople, Cases, Evidence, and Equipment

  To Employees;

  Grants Update On ArrestedPeople, PatrolRoutes, WorkingTimes, People, Officers, and Employees

  To Employees;

NOTES

- This Database is populated with a small amount of data compared to the size of actual police departments and to reach such the queries and database presented in this document would need to grow and become much more complex. This would include many more tables, people arrested people, and other relation data entries to be completely comprehensive and usable for an actual police department

- This database doesn't have the capabilities to address criminals who aren't arrested nor cases or evidence that lack an arrested person in the system. This system also lacks ways of keeping long term records of criminals who were arrested then moved to other facilities or released criminals from the department's holding.
- To address these concerns future versions of this database should have changes made to the cases table that allow for cases to not need to be linked directly to criminals arrested and being held in police custody, and to allow for those criminal's record to be updated so when they are released or moved to new facilities that errors do not occur in the database.

ISSUES, FUTURE CONSIDERATIONS