A Police Department Database

Jonathan Durbin

IST 659

Table of Contents

Summary

The goal of this project is to fully design and implement (using randomly generated data) a database that could be used by a (simplified understanding of a) police department (PD). Few institutions in the world likely have a consistent method of storing data – after all, the field of data science (and data storage) is relatively new. It is likely that a typical police department does not have a consistent data storage/retrieval solution in place, which is the motivation for this project.

NOTE: I used this link to contact the Auburn, NY PD in the interest of interviewing someone who understands the department's data needs.

There are 5 primary tables in this database, with 5 supporting associative tables. The 5 primary tables are Person, Warrant, Report, Weapon, and Vehicle. The list of tables and table relationships changed slightly after my interview with an Auburn PD (APD) Senior Clerk. The following database description is hopefully an improvement to what is currently in use at the APD. I learned in the interview that the APD has only four primary tables – Names,

Warrants/Wants, Reports, and Complaints/Incidents. In my (hopefully) improved database, I decided to combine the Reports and Complaints tables into one and add two tables to track information about vehicles and weapons.

Data Questions

1. How many reports of larceny are there?
2. What is the distribution of jobs in all the people in the database?
3. Which vehicles have weapons in them (and which weapons are they)?
4. What is the distribution of report types?
5. Find all Judges.

Business Rules

The business rules for the database designed in this project are as follows (entities are in **bold**):

- A **person** is defined as someone who has been in some form of contact with the PD. Officers, suspects, victims, etc. are on this list.
- A **weapon** is defined by its registration number and name (e.g., Glock 19).
- A **vehicle** is defined by its vehicle number, license, registration, and loadout.
    - A **vehicle** may hold one or more **weapons**.
- A **report** is defined by its subject(s), location(s), and content. (Subject(s) refers to the people who the report is primarily written about – suspects, felons, witnesses, victims, etc.)
    - A **person** may have written zero or more **reports**.
    - A **report** may be about any number of **persons**, **weapons**, or **vehicles**.
- A **warrant** is defined by the time it was signed by a judge, who signed it, and who it's for.

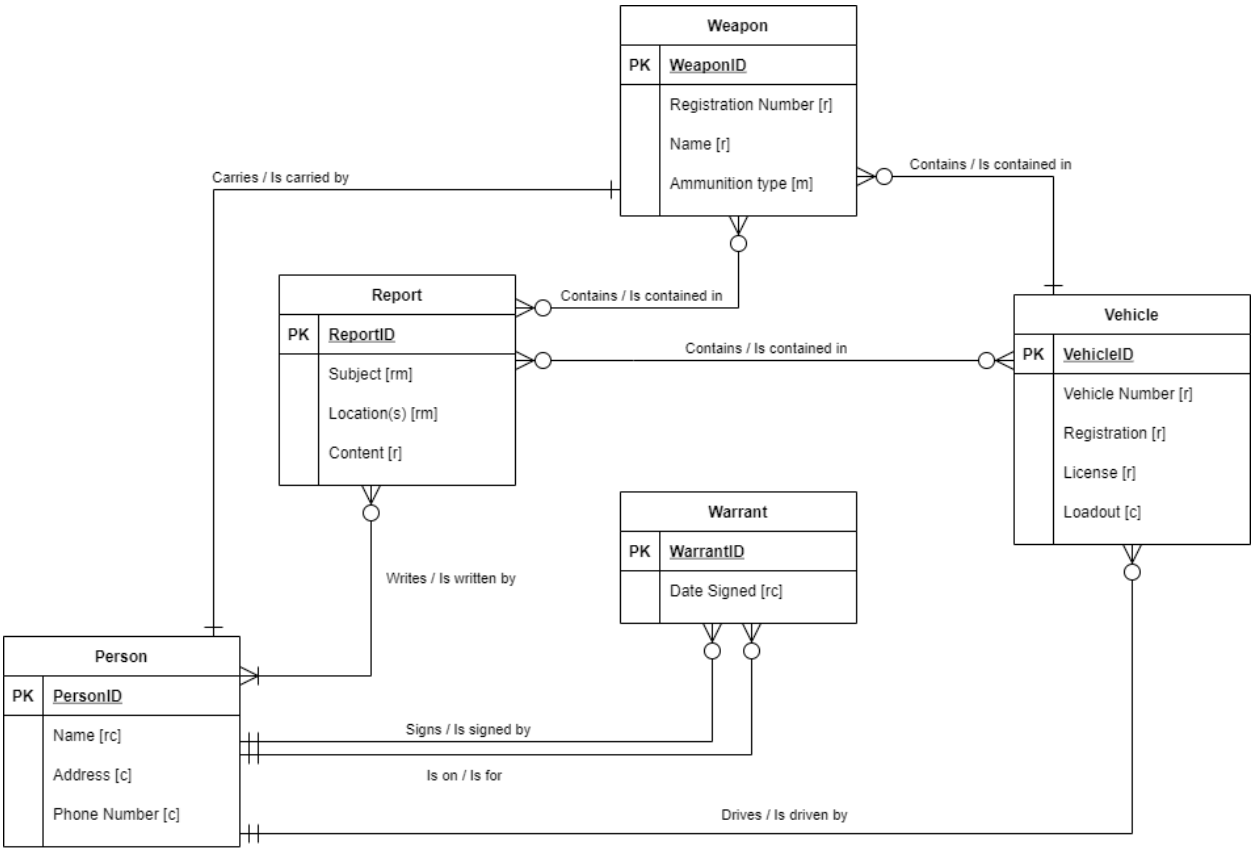
Stakeholders (what data do they need to access and maintain?)

The stakeholders in this case are those who work in the APD. Officers, Detectives, clerks, and other officials. Most stakeholders will likely be able to access all of the data, but only some can make edits. In other words, read only for everyone, but write privileges are reserved for the few who need them.

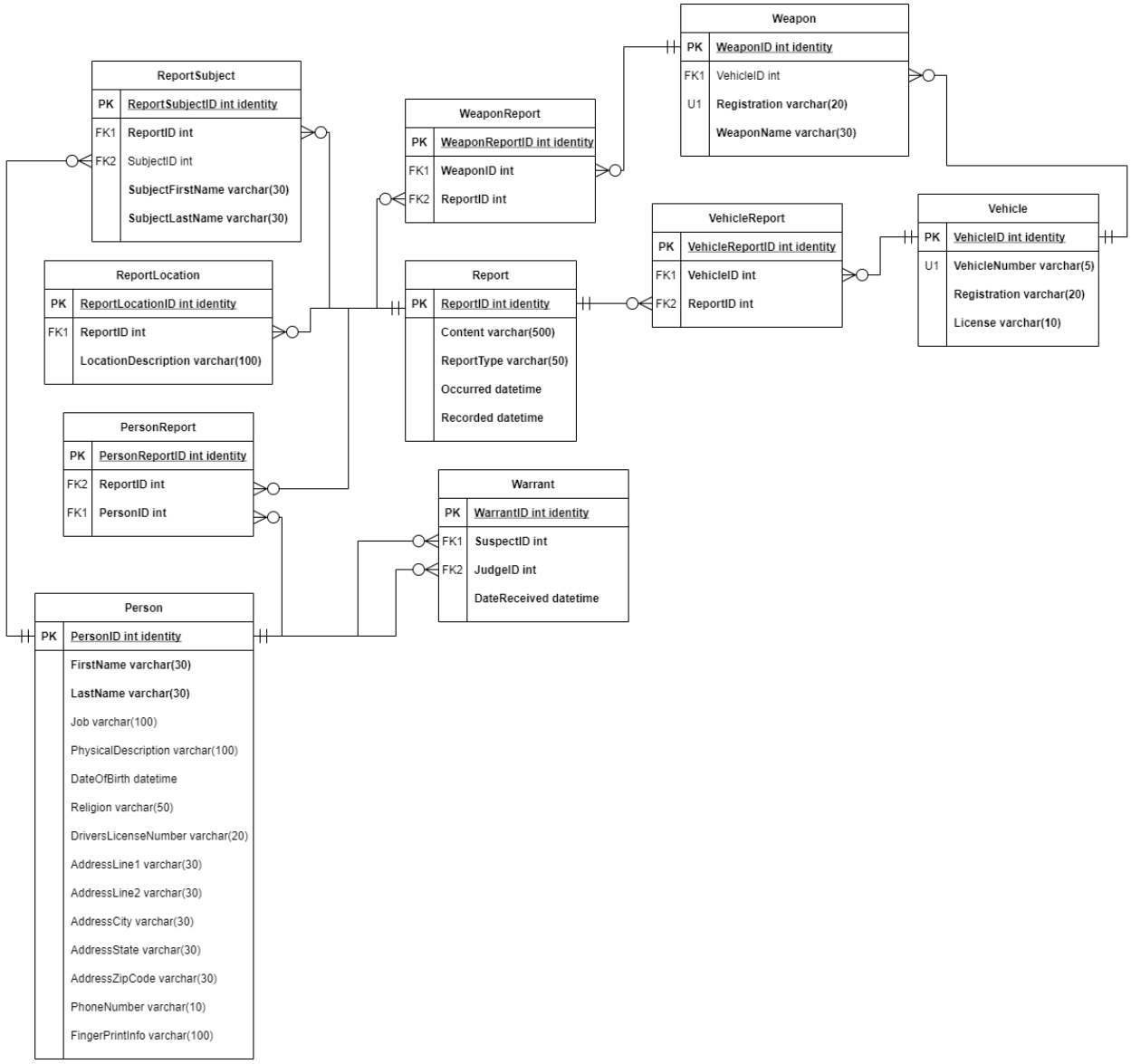
Glossary

- A Job is a short description of what a person does for a living. Officer, Judge, Clerk, and Bank Robber are just a few examples

Conceptual Model



Normalized Logical Model



Physical Database Design

In total, I have three functions (two of which are table-valued), three views, and 8 procedures. I would happily create more, but in the interest of finishing this project, I had to stop myself.

Below is a table of all of my design objects:

Name	Type
TotalPeople	FN
AddPerson	P
AddReport	P
AddVehicle	P
AddWarrant	P
AddWeapon	P
RemoveWarrant	P
UpdatePerson	P
UpdateReport	P
SearchCriminalWarrants	TF
SearchPerson	TF
AllReports	V
AllVehicles	V
AllVehiclesWeapons	V

Note: above list generated from:

```
SELECT
	[name],
	[type]
FROM sys.all_objects
WHERE
	(
	[type] = 'FN' OR
	[type] = 'IN' OR
	[type] = 'P' OR
	[type] = 'PC' OR
	[type] = 'TF' OR
	[type] = 'V' OR
	[type] = 'X'
	)
AND
[is_ms_shipped] = 0
ORDER BY [type], [name];
GO
```

Create Database

Shown in Appendix A – too much non-relevant text to paste into the body of my document.

Create Table

```
-- Dependent Tables
DROP TABLE IF EXISTS ReportSubject
DROP TABLE IF EXISTS ReportLocation
DROP TABLE IF EXISTS PersonReport
DROP TABLE IF EXISTS WeaponReport
DROP TABLE IF EXISTS VehicleReport
DROP TABLE IF EXISTS Warrant
DROP TABLE IF EXISTS Weapon

-- Independent Tables
DROP TABLE IF EXISTS Person
DROP TABLE IF EXISTS Report
DROP TABLE IF EXISTS Vehicle

GO

-- Independent Tables
CREATE TABLE Person (
	PersonID int identity,
	FirstName varchar(30) not null,
	LastName varchar(30) not null,
	Job varchar(100),
	PhysicalDescription varchar(100),
	DateOfBirth datetime,
	Religion varchar(50),
	DriversLicenseNumber varchar(20),
	AddressLine1 varchar(30),
	AddressLine2 varchar(30),
	AddressCity varchar(30),
	AddressState varchar(30),
	AddressZipCode varchar(30),
	PhoneNumber varchar(10),
	FingerPrintInfo varchar(100),
	CONSTRAINT PK_Person PRIMARY KEY (PersonID)
)

CREATE TABLE Report (
	ReportID int identity,
	Content varchar(500) not null,
	ReportType varchar(50) not null,
	Occurred datetime,
	Recorded datetime default GetDate()
	CONSTRAINT PK_Report PRIMARY KEY (ReportID)
)

CREATE TABLE Vehicle (
```

```
	VehicleID int identity,
	VehicleNumber varchar(5) not null,
	VehicleRegistration varchar(20) not null,
	License varchar(10) not null,
	CONSTRAINT PK_Vehicle PRIMARY KEY (VehicleID),
	CONSTRAINT U1_Vehicle UNIQUE (VehicleNumber)
)

-- Dependent Tables
CREATE TABLE Weapon (
	WeaponID int identity,
	VehicleID int,
	WeaponRegistration varchar(20) not null,
	WeaponName varchar(30) not null,
	CONSTRAINT PK_Weapon PRIMARY KEY (WeaponID),
	CONSTRAINT FK_Weapon_Vehicle FOREIGN KEY (VehicleID) REFERENCES
Vehicle(VehicleID),
	CONSTRAINT U1_Weapon UNIQUE (WeaponRegistration)
)

CREATE TABLE Warrant (
	WarrantID int identity,
	SuspectID int not null,
	JudgeID int not null,
	DateReceived datetime not null,
	CONSTRAINT PK_Warrant PRIMARY KEY (WarrantID),
	CONSTRAINT FK_Warrant_Suspect FOREIGN KEY (SuspectID) REFERENCES
Person(PersonID),
	CONSTRAINT FK_Warrant_Judge FOREIGN KEY (JudgeID) REFERENCES Person(PersonID)
)

CREATE TABLE VehicleReport (
	VehicleReportID int identity,
	VehicleID int not null,
	ReportID int not null,
	CONSTRAINT PK_VehicleReport PRIMARY KEY (VehicleReportID),
	CONSTRAINT FK_VehicleReport_Vehicle FOREIGN KEY (VehicleID) REFERENCES
Vehicle(VehicleID),
	CONSTRAINT FK_VehicleReport_Report FOREIGN KEY (ReportID) REFERENCES
Report(ReportID)
)

CREATE TABLE WeaponReport (
	WeaponReportID int identity,
	WeaponID int not null,
	ReportID int not null,
	CONSTRAINT PK_WeaponReport PRIMARY KEY (WeaponReportID),
	CONSTRAINT FK_WeaponReport_Weapon FOREIGN KEY (WeaponID) REFERENCES
Weapon(WeaponID),
	CONSTRAINT FK_WeaponReport_Report FOREIGN KEY (ReportID) REFERENCES
Report(ReportID)
)

CREATE TABLE PersonReport (
	PersonReportID int identity,
```

```
	PersonID int not null,
	ReportID int not null,
	CONSTRAINT PK_PersonReport PRIMARY KEY (PersonReportID),
	CONSTRAINT FK_PersonReport_Person FOREIGN KEY (PersonID) REFERENCES
Person(PersonID),
	CONSTRAINT FK_PersonReport_Report FOREIGN KEY (ReportID) REFERENCES
Report(ReportID)
)

CREATE TABLE ReportLocation (
	ReportLocationID int identity,
	ReportID int not null,
	LocationDescription varchar(100) not null,
	CONSTRAINT PK_ReportLocation PRIMARY KEY (ReportLocationID),
	CONSTRAINT FK_ReportLocation FOREIGN KEY (ReportID) REFERENCES
Report(ReportID)
)

CREATE TABLE ReportSubject (
	ReportSubjectID int identity,
	ReportID int not null,
	SubjectID int,
	SubjectFirstName varchar(30) not null,
	SubjectLastName varchar(30) not null,
	CONSTRAINT PK_ReportSubject PRIMARY KEY (ReportSubjectID),
	CONSTRAINT FK_ReportSubject_Report FOREIGN KEY (ReportID) REFERENCES
Report(ReportID),
	CONSTRAINT FK_ReportSubject_Person FOREIGN KEY (SubjectID) REFERENCES
Person(PersonID)
)

GO
```

Create Procedure

```
/*
Procedures:
	Update a report
	Create a new report
	Update a Person's record
	Create a new Person record
	Add a new vehicle
	Add a new weapon
	Add a new warrant
	Remove a warrant
*/

-- Function to update a report after it's been written.
--   (or while it's being written for the first time)
CREATE OR ALTER PROCEDURE UpdateReport(
```

```
	@ReportID int,
	@PersonResponsible int,
	@SubjectFirstName varchar(30) = NULL,
	@SubjectLastName varchar(30) = NULL,
	@Location varchar(100) = NULL,
	@WeaponRegistration varchar(20) = NULL,
	@VehicleNumber varchar(5) = NULL
	) AS
BEGIN
	-- update ReportSubject Table
	IF (@SubjectFirstName IS NOT NULL OR @SubjectLastName IS NOT NULL)
		BEGIN
			DECLARE @s_id int
			-- Assume the firstname, lastname combination for the subject
accurately matches
			--   the correct name in the persons database.
			-- This is a naive assumption - some people have the same name.
			SELECT @s_id = PersonID
			FROM Person
			WHERE
				FirstName = @SubjectFirstName AND
				LastName = @SubjectLastName

			INSERT INTO ReportSubject
				(ReportID, SubjectID, SubjectFirstName, SubjectLastName)
			-- SubjectID could be null, but that's okay.
			-- Not all people will be in the Person table.
			VALUES
				(@ReportID, @s_id, @SubjectFirstName, @SubjectLastName)
		END

	-- Update ReportLocation Table
	IF (@Location IS NOT NULL)
		BEGIN
			INSERT INTO ReportLocation
				(ReportID, LocationDescription)
			VALUES
				(@ReportID, @Location)
		END

	-- Update WeaponReport Table
	IF (@WeaponRegistration IS NOT NULL)
		BEGIN
			DECLARE @w_id int
			SELECT @w_id = WeaponID
			FROM Weapon
			WHERE WeaponRegistration = @WeaponRegistration

			INSERT INTO WeaponReport
				(WeaponID, ReportID)
			VALUES
				(@w_id, @ReportID)
		END

	-- Update VehicleReport Table
```

```
	IF (@VehicleNumber IS NOT NULL)
		BEGIN
			DECLARE @v_id int
			SELECT @v_id = VehicleID
			FROM Vehicle
			WHERE VehicleNumber = @VehicleNumber

			INSERT INTO VehicleReport
				(VehicleID, ReportID)
			VALUES
				(@v_id, @ReportID)
		END

	-- Update PersonReport Table (Person updating the report)
	INSERT INTO PersonReport
		(ReportID, PersonID)
	VALUES
		(@ReportID, @PersonResponsible)
END
GO

-- Create a new report - the user can specify as much information as they have now
--   or come back at a later time and update the report with "UpdateReport"
CREATE OR ALTER PROCEDURE AddReport(
	@Content varchar(500),
	@ReportType varchar(50),
	@Occurred datetime,
	@PersonResponsible int,
	@SubjectFirstName varchar(30) = NULL,
	@SubjectLastName varchar(30) = NULL,
	@Location varchar(100) = NULL,
	@WeaponRegistration varchar(20) = NULL,
	@VehicleNumber varchar(5) = NULL
	) AS
BEGIN
	-- Update the Report table with the body of the report
	INSERT INTO Report (Content, ReportType, Occurred)
	VALUES (@Content, @ReportType, @Occurred)

	DECLARE @s_id int
	SELECT @s_id = SCOPE_IDENTITY()

	-- Update the associative tables with the relevant information
	EXEC UpdateReport
		@ReportID = @s_id,
		@PersonResponsible = @PersonResponsible,
		@SubjectFirstName = @SubjectFirstName,
		@SubjectLastName = @SubjectLastName,
		@Location = @Location,
		@WeaponRegistration = @WeaponRegistration,
		@VehicleNumber = @VehicleNumber
END
GO
```

```
-- Update a Person's information
CREATE OR ALTER PROCEDURE UpdatePerson (
	@PersonID int,
	@Job varchar(100) = NULL,
	@PhysicalDescription varchar(100) = NULL,
	@DateOfBirth datetime = NULL,
	@Religion varchar(50) = NULL,
	@DriversLicenseNumber varchar(20) = NULL,
	@AddressLine1 varchar(30) = NULL,
	@AddressLine2 varchar(30) = NULL,
	@AddressCity varchar(30) = NULL,
	@AddressState varchar(30) = NULL,
	@AddressZipCode varchar(30) = NULL,
	@PhoneNumber varchar(10) = NULL,
	@FingerPrintInfo varchar(100) = NULL
	) AS
BEGIN
	-- This seems repetive, but I think I need to check
	--   each parameter individually
	--   because I don't want to update a row with null values
	--   when it already has non-null values in it.
    IF (@Job IS NOT NULL)
        UPDATE Person SET Job = @Job WHERE PersonID = @PersonID
    IF (@PhysicalDescription IS NOT NULL)
        UPDATE Person SET PhysicalDescription = @PhysicalDescription WHERE PersonID =
@PersonID
    IF (@DateOfBirth IS NOT NULL)
        UPDATE Person SET DateOfBirth = @DateOfBirth WHERE PersonID = @PersonID
    IF (@Religion IS NOT NULL)
        UPDATE Person SET Religion = @Religion WHERE PersonID = @PersonID
    IF (@DriversLicenseNumber IS NOT NULL)
        UPDATE Person SET DriversLicenseNumber = @DriversLicenseNumber WHERE PersonID
= @PersonID
    IF (@AddressLine1 IS NOT NULL)
        UPDATE Person SET AddressLine1 = @AddressLine1 WHERE PersonID = @PersonID
    IF (@AddressLine2 IS NOT NULL)
        UPDATE Person SET AddressLine2 = @AddressLine2 WHERE PersonID = @PersonID
    IF (@AddressCity IS NOT NULL)
        UPDATE Person SET AddressCity = @AddressCity WHERE PersonID = @PersonID
    IF (@AddressState IS NOT NULL)
        UPDATE Person SET AddressState = @AddressState WHERE PersonID = @PersonID
    IF (@AddressZipCode IS NOT NULL)
        UPDATE Person SET AddressZipCode = @AddressZipCode WHERE PersonID = @PersonID
    IF (@PhoneNumber IS NOT NULL)
        UPDATE Person SET PhoneNumber = @PhoneNumber WHERE PersonID = @PersonID
    IF (@FingerPrintInfo IS NOT NULL)
        UPDATE Person SET FingerPrintInfo = @FingerPrintInfo WHERE PersonID =
@PersonID
END
GO

-- Create a new Person record
CREATE OR ALTER PROCEDURE AddPerson (
	@FirstName varchar(30),
	@LastName varchar(30),
```

```
	@Job varchar(100) = NULL,
	@PhysicalDescription varchar(100) = NULL,
	@DateOfBirth datetime = NULL,
	@Religion varchar(50) = NULL,
	@DriversLicenseNumber varchar(20) = NULL,
	@AddressLine1 varchar(30) = NULL,
	@AddressLine2 varchar(30) = NULL,
	@AddressCity varchar(30) = NULL,
	@AddressState varchar(30) = NULL,
	@AddressZipCode varchar(30) = NULL,
	@PhoneNumber varchar(10) = NULL,
	@FingerPrintInfo varchar(100) = NULL
	) AS
BEGIN
    INSERT INTO Person (FirstName, LastName)
    VALUES (@FirstName, @LastName)

	DECLARE @s_id int
	SELECT @s_id = SCOPE_IDENTITY()

	-- Fill in the rest of the information as provided (if provided)
	EXEC UpdatePerson
		@PersonID = @s_id,
		@Job = @Job,
		@PhysicalDescription = @PhysicalDescription,
		@DateOfBirth = @DateOfBirth,
		@Religion = @Religion,
		@DriversLicenseNumber = @DriversLicenseNumber,
		@AddressLine1 = @AddressLine1,
		@AddressLine2 = @AddressLine2,
		@AddressCity = @AddressCity,
		@AddressState = @AddressState,
		@AddressZipCode = @AddressZipCode,
		@PhoneNumber = @PhoneNumber,
		@FingerPrintInfo = @FingerPrintInfo
END
GO


-- Add new vehicle
CREATE OR ALTER PROCEDURE AddVehicle (
	@VehicleNumber varchar(5),
	@Registration varchar(20),
	@License varchar(10)
	) AS
BEGIN
	INSERT INTO Vehicle (VehicleNumber, VehicleRegistration, License)
	VALUES (@VehicleNumber, @Registration, @License)
END
GO


-- Add new weapon
CREATE OR ALTER PROCEDURE AddWeapon (
	@VehicleID int = NULL,
```

```
	@Registration varchar(20),
	@WeaponName varchar(30)
	) AS
BEGIN
	IF (@VehicleID IS NULL)
	BEGIN
		INSERT INTO Weapon (WeaponRegistration, WeaponName)
		VALUES (@Registration, @WeaponName)
	END
	ELSE
	BEGIN
		INSERT INTO Weapon (VehicleID, WeaponRegistration, WeaponName)
		VALUES (@VehicleID, @Registration, @WeaponName)
	END
END
GO


-- Add a new warrant
CREATE OR ALTER PROCEDURE AddWarrant (
	@SuspectID int,
	@JudgeID int,
	@DateReceived datetime
	) AS
BEGIN
	INSERT INTO Warrant (SuspectID, JudgeID, DateReceived)
	VALUES (@SuspectID, @JudgeID, @DateReceived)
END
GO

-- Remove a warrant
CREATE OR ALTER PROCEDURE RemoveWarrant (
	@WarrantID int
	) AS
BEGIN
	DELETE FROM Warrant
	WHERE WarrantID = @WarrantID
END
GO
```

Create Function

```
-- Get person's record by first name / last name / DOB
CREATE OR ALTER FUNCTION dbo.SearchPerson (
	@PersonID int = NULL,
	@FirstName varchar(30) = NULL,
	@LastName varchar(30) = NULL,
	@DateOfBirth datetime = NULL
	)
	RETURNS @returnPerson TABLE (
		PersonID int,
```

```
		FirstName varchar(30),
		LastName varchar(30),
		Job varchar(100),
		PhysicalDescription varchar(100),
		DateOfBirth datetime,
		Religion varchar(50),
		DriversLicenseNumber varchar(20),
		AddressLine1 varchar(30),
		AddressLine2 varchar(30),
		AddressCity varchar(30),
		AddressState varchar(30),
		AddressZipCode varchar(30),
		PhoneNumber varchar(10),
		FingerPrintInfo varchar(100)
	)
AS
BEGIN
	INSERT INTO @returnPerson
	SELECT * FROM Person
	WHERE
		PersonID = @PersonID OR
		FirstName = @FirstName OR
		LastName = @LastName OR
		DateOfBirth = @DateOfBirth
	RETURN
END
GO

-- Get warrants for a person (connected to dbo.SearchPerson)
CREATE OR ALTER FUNCTION SearchCriminalWarrants (
	@PersonID int = NULL,
	@FirstName varchar(30) = NULL,
	@LastName varchar(30) = NULL,
	@DateOfBirth datetime = NULL
	)
	RETURNS @returnWarrants TABLE (
		WarrantID int,
		SuspectID int,
		JudgeID int,
		DateReceived datetime
	)
AS
BEGIN
	-- List of person ID's
	DECLARE @person TABLE (
		p_id int
	)
	-- Fill with values from dbo.SearchPerson function
	INSERT INTO @person
		SELECT PersonID
		FROM dbo.SearchPerson(
			default,
			@FirstName,
			@LastName,
			@DateOfBirth
```

```
		)

	INSERT INTO @returnWarrants
	SELECT *
	FROM Warrant
	WHERE
		SuspectID = @PersonID OR
		SuspectID IN (SELECT * FROM @person)
	RETURN
END
GO


-- Total Number of People
CREATE OR ALTER FUNCTION dbo.TotalPeople ()
RETURNS int AS
BEGIN
	DECLARE @returnValue int
	SELECT @returnValue = COUNT(PersonID) FROM Person
	RETURN @returnValue
END
GO


-- Look up a person's address given their ID
CREATE OR ALTER FUNCTION dbo.GetPersonAddress (@PersonID int)
RETURNS varchar(150) AS
BEGIN
	DECLARE @returnValue varchar(150)
	SELECT
		@returnValue = CONCAT(
			AddressLine1, ' ',
			AddressLine2, ' ',
			AddressCity, ', ',
			AddressState, ' ',
			AddressZipCode
		)
	FROM Person
	WHERE PersonID = @PersonID

	RETURN @returnValue
END
GO
```

Create View

```
-- View: All Vehicles
-- View: All Weapons / Assigned Vehicle
-- View: All reports
-- View: All Warrants
```

```sql
CREATE OR ALTER VIEW AllVehicles AS
	SELECT *
	FROM Vehicle
GO

CREATE OR ALTER VIEW AllVehiclesWeapons AS
	SELECT
		Vehicle.VehicleID,
		Weapon.WeaponRegistration,
		Weapon.WeaponName,
		Vehicle.VehicleNumber,
		Vehicle.VehicleRegistration,
		Vehicle.License
	FROM Vehicle
	LEFT JOIN Weapon ON Vehicle.VehicleID = Weapon.VehicleID
	WHERE WeaponRegistration IS NOT NULL AND WeaponName IS NOT NULL
GO


CREATE OR ALTER VIEW AllReports AS
	SELECT
		Report.ReportID,
		Report.ReportType,
		Person.PersonID AS PersonResponsibleID,
		Person.FirstName AS PersonResponsibleFirstName,
		Person.LastName AS PersonResponsibleLastName,
		ReportSubject.SubjectFirstName,
		ReportSubject.SubjectLastName,
		ReportLocation.LocationDescription,
		Weapon.WeaponName,
		Weapon.WeaponRegistration,
		Vehicle.VehicleNumber AS RespondingVehicle,
		Report.Content
	FROM Report
	LEFT JOIN ReportLocation ON ReportLocation.ReportID = Report.ReportID
	LEFT JOIN ReportSubject ON ReportSubject.ReportID = Report.ReportID
	LEFT JOIN VehicleReport ON VehicleReport.ReportID = Report.ReportID
		LEFT JOIN Vehicle ON VehicleReport.VehicleID = Vehicle.VehicleID
	LEFT JOIN WeaponReport ON WeaponReport.ReportID = Report.ReportID
		LEFT JOIN Weapon ON WeaponReport.WeaponID = Weapon.WeaponID
	LEFT JOIN PersonReport ON PersonReport.ReportID = Report.ReportID
		LEFT JOIN Person ON PersonReport.PersonID = Person.PersonID
GO

CREATE OR ALTER VIEW AllWarrants AS
	SELECT
		WarrantID,
		CONVERT(varchar(20), CONVERT(date, DateReceived)) AS DateReceived,
		JudgeID,
		CONCAT(Judge.FirstName, ' ', Judge.Lastname) AS [Judge Name],
		SuspectID,
		CONCAT(Suspect.FirstName, ' ', Suspect.Lastname) AS [Suspect Name],
		Suspect.Job,
		Suspect.PhysicalDescription,
		Suspect.DateOfBirth,
```

```
            Suspect.Religion,
            Suspect.DriversLicenseNumber,
            dbo.GetPersonAddress(Suspect.PersonID) AS [Suspect Address],
            Suspect.PhoneNumber,
            Suspect.FingerPrintInfo
        FROM Warrant
        JOIN Person as Judge
        ON Warrant.JudgeID = Judge.PersonID
        JOIN Person as Suspect
        ON Warrant.SuspectID = Suspect.PersonID
GO
```

Data Creation

I used procedures to enter my data – inserting a report, for example involves potentially inserting into multiple tables, so I think it saves space and time. Some representative statements are below, along with the parameters for the corresponding stored procedures for reference:

Add Vehicles

- @VehicleNumber varchar(5)
- @Registration varchar(20)
- @License varchar(10)

```
EXEC AddVehicle '1', 'pszz 1110', '1503744644230065'
EXEC AddVehicle '2', 'kmvj 2271', '2250883486568355'
EXEC AddVehicle '3', 'xefo 0837', '2344326554236184'
EXEC AddVehicle '4', 'hggy 0784', '7572903483447640'
EXEC AddVehicle '5', 'wbcr 5034', '3123055388946135'
```

Add Weapons

- @VehicleID int (default is NULL)
- @Registration varchar(20)
- @WeaponName varchar(30)

```
EXEC AddWeapon default, '1', 'Akdal Ghost TR01'
EXEC AddWeapon default, '2', 'ALFA Combat'
EXEC AddWeapon default, '3', 'ALFA Defender'
EXEC AddWeapon default, '4', '9mm Winchester Magnum'
EXEC AddWeapon default, '5', 'AMT AutoMag IV'
EXEC AddWeapon 1, '6', 'AMT Automag V'
EXEC AddWeapon 1, '7', 'AMT Backup'
```

Add Persons

- @FirstName varchar(30)
- @LastName varchar(30)
- @Job varchar(100) (default is NULL)
- @PhysicalDescription varchar(100) (default is NULL)
- @DateOfBirth datetime (default is NULL)
- @Religion varchar(50) (default is NULL)
- @DriversLicenseNumber varchar(20) (default is NULL)
- @AddressLine1 varchar(30) (default is NULL)
- @AddressLine2 varchar(30) (default is NULL)
- @AddressCity varchar(30) (default is NULL)
- @AddressState varchar(30) (default is NULL)
- @AddressZipCode varchar(30) (default is NULL)
- @PhoneNumber varchar(10) (default is NULL)
- @FingerPrintInfo varchar(100) (default is NULL)

```
EXEC AddPerson 'Vanda','Newall','Data
Scientist','Small','2015/03/23','Agnostic','569556455603898','2760 Hayes
Parkway',default,'Auburn','NY','12365','6814672251','3l5vuo708ju58o8'
EXEC AddPerson
'Grant','Pellingar',default,'Blue','1960/08/28','Agnostic','692672501682064','0 Moose
Center',default,'Auburn','NY','12385','7066687750','0k5jli081ir12n3'
EXEC AddPerson
'Gualterio','Beadnell','Officer','Big','1974/04/03',default,'167891409895782','1
Forster Point',default,'Auburn','NY','12405','7744165654','0l2mhl132kp60i4'
EXEC AddPerson
'Layla','Pleasance','Clerk',default,'1995/06/17',default,'440931417973592','12901
Bowman Avenue',default,'Auburn','NY','12425','4409776672','3x8ndm983ki96i4'
EXEC AddPerson 'Annabela','Pringell','Data
Scientist','Big','2003/01/02','Christian','925864814908899','64773 Bunting
Point',default,'Auburn','NY','12445','6137248473','9p9yjh373ij93h3'
EXEC AddPerson
'Courtney','Gaffon','Judge','Big','1966/06/22','Agnostic','699247080776805','20303
Lighthouse Bay Drive',default,'Auburn','NY','12465','6621136045','1n3gfx024bf95x1'
```

Add Reports

- @Content varchar(500)
- @ReportType varchar(50)
- @Occurred datetime
- @PersonResponsible int
- @SubjectFirstName varchar(30) (default is NULL)
- @SubjectLastName varchar(30) (default is NULL)
- @Location varchar(100) (default is NULL)
- @WeaponRegistration varchar(20) (default is NULL)
- @VehicleNumber varchar(5) (default is NULL)

```
EXEC AddReport 'duis bibendum morbi non quam nec dui luctus rutrum nulla','Criminal
Mischief','6/16/1951',1,'Marney','Neath','2140 Vernon Park','1','1'
```

```
EXEC AddReport 'libero nullam sit amet turpis elementum ligula vehicula consequat morbi','Criminal Mischief','10/17/1987',2,'Lilian','Cordelette','174 Forest Alley','2','2'
EXEC AddReport 'sapien arcu sed augue aliquam erat volutpat in congue etiam','Criminal Mischief','4/22/1990',3,'Norris','Senchenko','581 Dorton Crossing','3','3'
EXEC AddReport 'cubilia curae duis faucibus accumsan odio curabitur convallis duis consequat','Larceny','12/3/1999',4,'Terrel','Millom','75325 Goodland Trail','4','4'
EXEC AddReport 'lorem quisque ut erat curabitur gravida nisi at nibh in','Armed Robbery','1/12/1965',5,'Caril','Ferrierio','0 Birchwood Circle','5','5'
```

Data Manipulation

An example of using one of my stored procedures for data manipulation is the following situation:

A report that was filed yesterday was missing a subject, as they could not be tracked down until today. Using the procedure UpdateReport, a clerk runs the following statement, knowing the parameters (included below for convenience).

- @ReportID int
- @PersonResponsible int
- @SubjectFirstName varchar(30) (default is NULL)
- @SubjectLastName varchar(30) (default is NULL)
- @Location varchar(100) (default is NULL)
- @WeaponRegistration varchar(20) (default is NULL)
- @VehicleNumber varchar(5) (default is NULL)

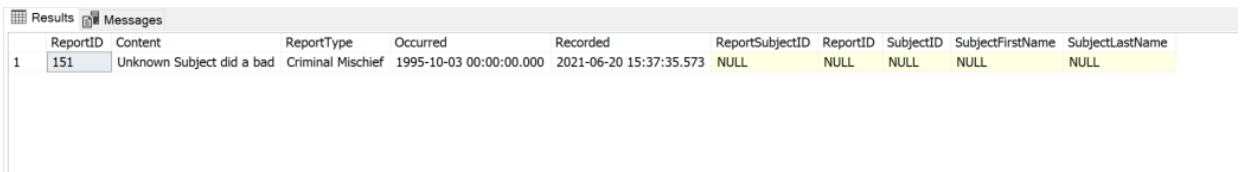
This ran yesterday:

```
EXEC AddReport 'Unknown Subject did a bad','Criminal Mischief','10/3/1995',10,default,default,'93446 Saint Paul Alley',default,'50'
GO
```

This ran today:

```
SELECT
	*
FROM Report
LEFT JOIN ReportSubject ON ReportSubject.ReportID = Report.ReportID
WHERE Occurred = '10/3/1994'
GO
```

Results:



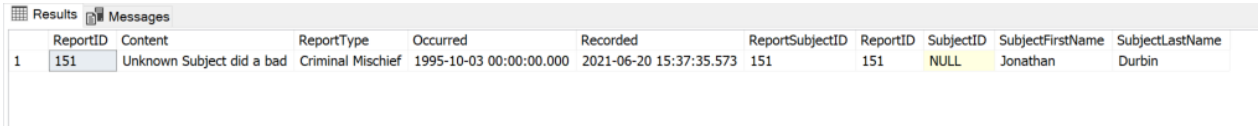
Results | Messages

	ReportID	Content	ReportType	Occurred	Recorded	ReportSubjectID	ReportID	SubjectID	SubjectFirstName	SubjectLastName
1	151	Unknown Subject did a bad	Criminal Mischief	1995-10-03 00:00:00.000	2021-06-20 15:37:35.573	NULL	NULL	NULL	NULL	NULL

Updating the report:

```
EXEC UpdateReport 151, 20, 'Jonathan', 'Durbin', default, default, default
GO
```

Results:



Results Messages

	ReportID	Content	ReportType	Occurred	Recorded	ReportSubjectID	ReportID	SubjectID	SubjectFirstName	SubjectLastName
1	151	Unknown Subject did a bad	Criminal Mischief	1995-10-03 00:00:00.000	2021-06-20 15:37:35.573	151	151	NULL	Jonathan	Durbin

Answering Data Questions

If I could spend all day coming up with interesting data questions, I would. However, I'll have to settle with what I can come up with in this moment.

How many reports of larceny are there?

```
SELECT COUNT(ReportID) FROM AllReports WHERE ReportType = 'Larceny'
```

	(No column name)
1	41

What is the distribution of jobs in all the people in the database?

```
SELECT
	COUNT(*) AS [Total Jobs],
	Job
FROM Person
GROUP BY Job
```

	Total Jobs	Job
1	45	NULL
2	35	Clerk
3	40	Construction Worker
4	47	Data Scientist
5	34	Judge
6	49	Officer

Which vehicles have weapons in them (and which weapons are they)?

```
SELECT
	Vehicle.VehicleID,
	WeaponID,
	WeaponName
FROM Vehicle
JOIN Weapon ON Vehicle.VehicleID = Weapon.VehicleID
```

	VehicleID	WeaponID	WeaponName
1	1	6	AMT Automag V
2	1	7	AMT Backup
3	2	42	Berloque pistol
4	2	43	Kingdom of Hungary
5	3	99	Glisenti Model 1910
6	3	100	Glock 36
7	3	101	Glock 37

What is the distribution of report types?

```
SELECT
	COUNT(*) AS [Total Report Types],
	ReportType
FROM Report
GROUP BY ReportType
```

	Total Report Types	ReportType
1	25	Armed Robbery
2	30	Assault
3	55	Criminal Mischief
4	41	Larceny

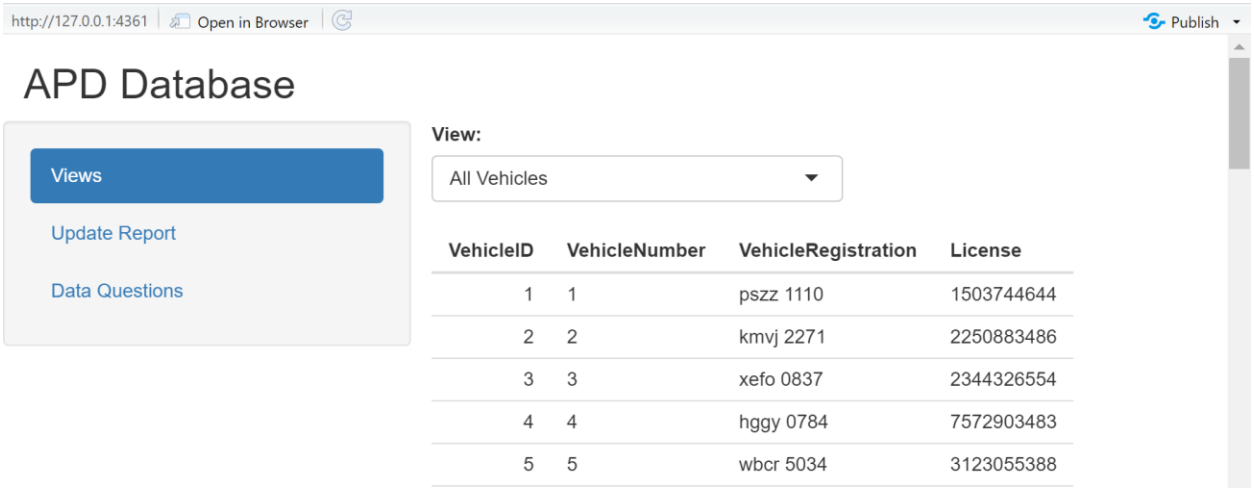
Find all Judges.

```
SELECT * FROM Person WHERE Job = 'Judge'
```

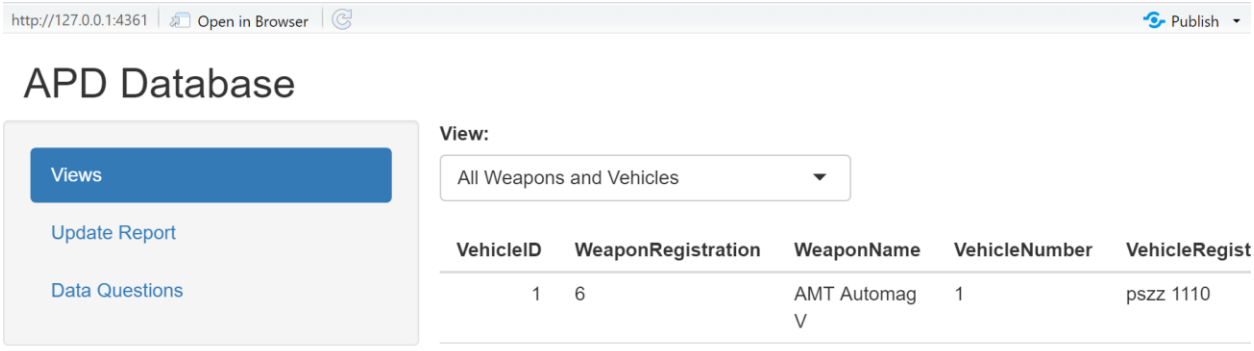
	PersonID	FirstName	LastName	Job	PhysicalDescription	DateOfBirth	Religion	DriversLicenseNumber	AddressLine1	AddressLine2	AddressCity	AddressState	AddressZipCode	PhoneNumber	FingerPrintInfo
1	6	Courtney	Gaffon	Judge	Big	1966-06-22 00:00:00.000	Agnostic	699247080776805	20303 Lighthouse Bay Drive	NULL	Auburn	NY	12465	6621136045	1n3gfx024bf95x1
2	12	Lainey	Vickar	Judge	Small	1983-03-16 00:00:00.000	Christian	975479854912172	720 Summerview Alley	NULL	Auburn	NY	12585	8857777990	5c5pyk039bj65s3
3	14	Terence	Beste	Judge	Big	1988-07-13 00:00:00.000	Muslim	587252046306238	34 Rieder Hill	NULL	Auburn	NY	12625	6841201988	1n1pcd547sk5...
4	17	Barthol...	Matus	Judge	Purple	1959-08-26 00:00:00.000	Christian	464394970019823	9 Anzinger Point	NULL	Auburn	NY	12685	1709771836	5n4mpn996hj5...
5	18	Bekki	Gomez	Judge	Small	2005-02-12 00:00:00.000	Agnostic	704324130330030	604 Oakridge Lane	NULL	Auburn	NY	12705	5747544387	0n8fhq266hr93...
6	19	Archy	Berrigan	Judge	NULL	2014-08-27 00:00:00.000	Buddhist	224906288478648	933 Elgar Hill	NULL	Auburn	NY	12725	5579940201	3p7lfo119xq94...
7	22	Darell	Houlton	Judge	Purple	1989-03-03 00:00:00.000	Buddhist	298857450193883	85 Surrey Terrace	NULL	Auburn	NY	12785	1396502272	9c6qcx066vg1...
8	23	Andeee	Osipenko	Judge	Blue	2018-04-24 00:00:00.000	Buddhist	944606194622280	69444 Lawn Pass	NULL	Auburn	NY	12805	4013360483	2z7aeq622ex8...
9	26	Livia	Jolley	Judge	NULL	NULL	Christian	793507569453550	23141 Nova Court	NULL	Auburn	NY	12865	7366006160	4z4wyu632th6...
10	32	Fawne	McLauc...	Judge	Blue	1998-04-19 00:00:00.000	NULL	097251575551478	916 Mccormick Road	NULL	Auburn	NY	12985	6296122778	8f7emj655rn56...
11	48	Harrietta	Saunton	Judge	Big	1981-10-11 00:00:00.000	Agnostic	088725398731948	5 Delaware Avenue	NULL	Auburn	NY	13305	3344058711	1r0yhe494bz1...
12	64	Rolf	Grovier	Judge	NULL	2014-03-30 00:00:00.000	NULL	581412972224289	8 Fallview Hill	NULL	Auburn	NY	13625	8731335429	8c2zhg134ud6...
13	68	Hilton	Wain	Judge	Purple	2003-02-24 00:00:00.000	NULL	883543415397674	46456 Farmco Terrace	NULL	Auburn	NY	13705	3281169957	2a4yoa194wk8...

Implementation

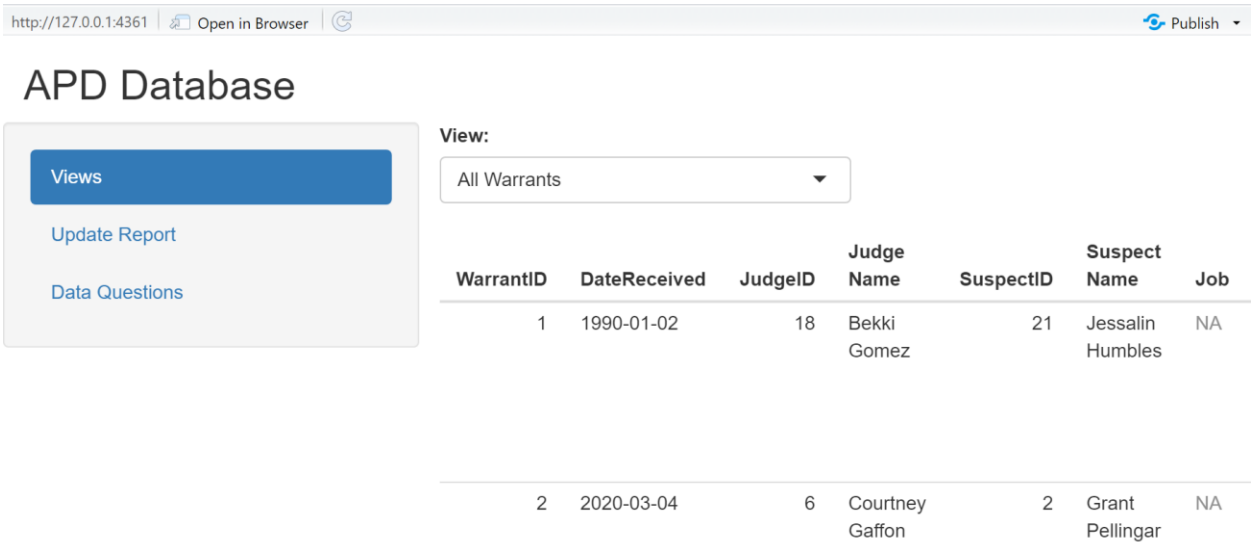
Views:



VehicleID	VehicleNumber	VehicleRegistration	License
1	1	pszz 1110	1503744644
2	2	kmvj 2271	2250883486
3	3	xefo 0837	2344326554
4	4	hggy 0784	7572903483
5	5	wbcr 5034	3123055388



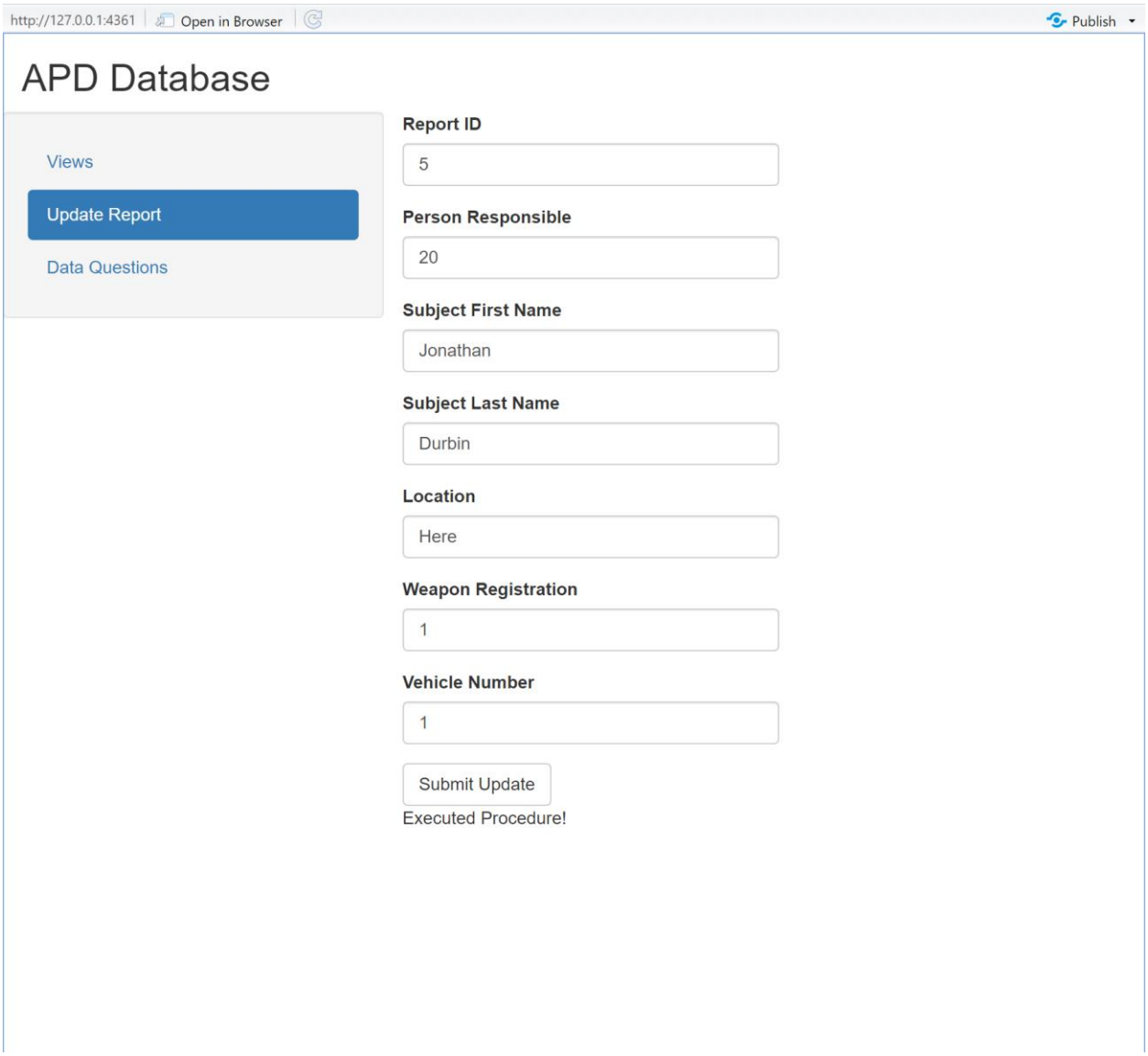
VehicleID	WeaponRegistration	WeaponName	VehicleNumber	VehicleRegist
1	6	AMT Automag V	1	pszz 1110
1	7	AMT Backup	1	pszz 1110
2	42	Berloque pistol	2	kmvj 2271
2	43	Kingdom of Hungary	2	kmvj 2271
3	99	Glisenti Model 1910	3	xefo 0837
3	100	Glock 36	3	xefo 0837
3	101	Glock 37	3	xefo 0837



Update Report

Before:

Values entered:

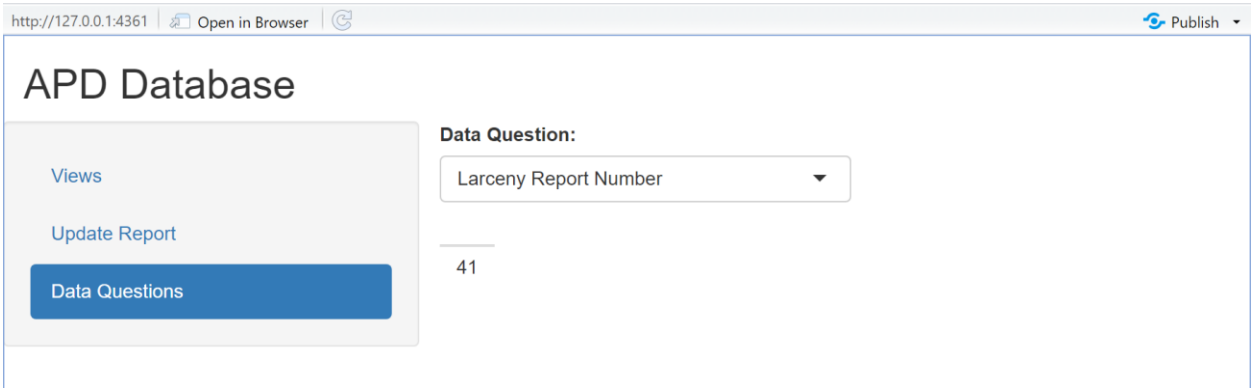


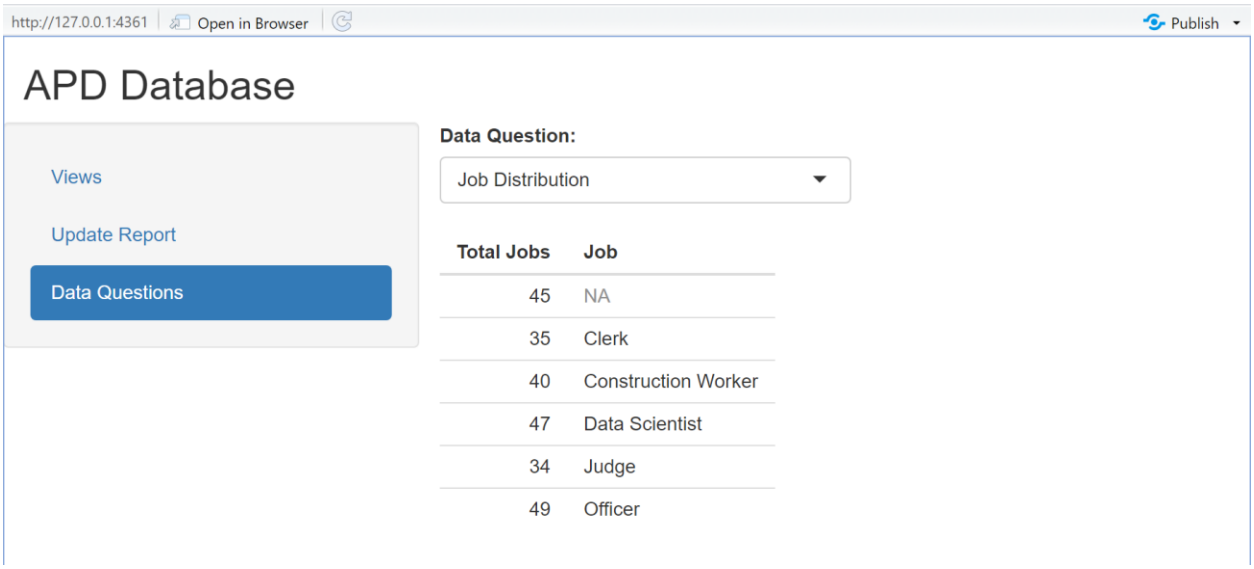
After:

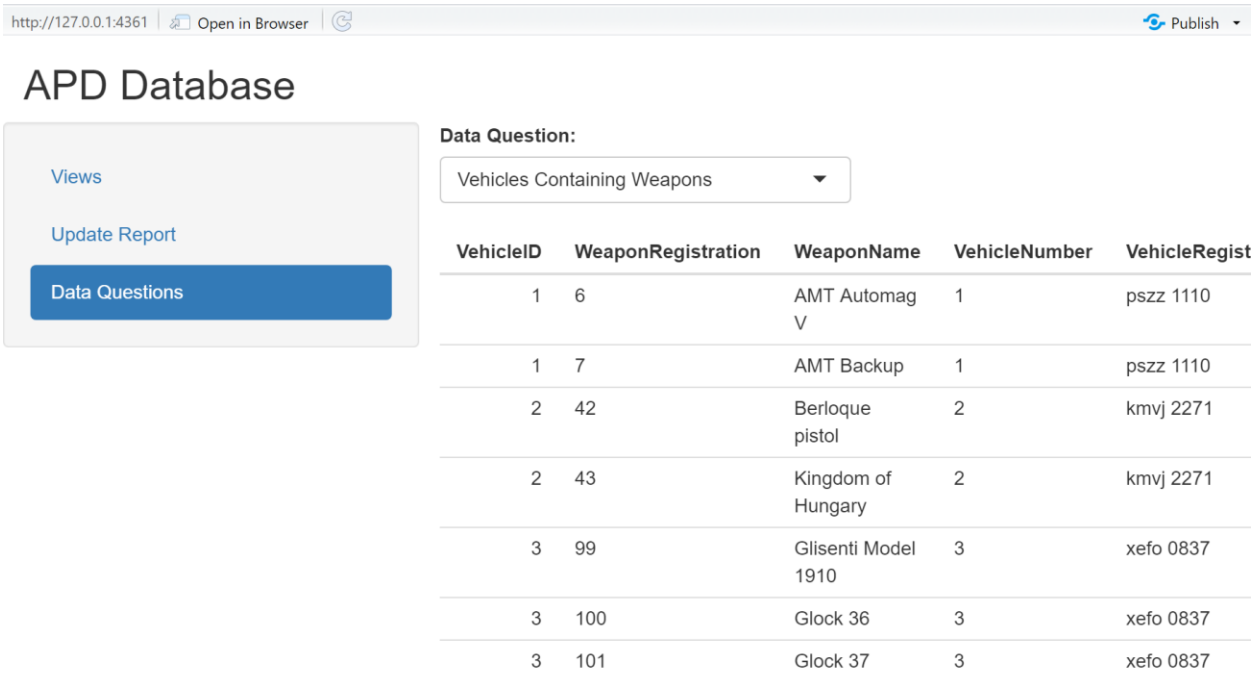
Results Messages

	ReportID	Content	ReportType	Occurred	Recorded	ReportSubjectID	ReportID	SubjectID	SubjectFirstName	SubjectLastName
1	5	lorem quisque ut erat curabitur gravida nisi at...	Armed Robbery	1965-01-12 00:00:00.000	2021-06-20 15:36:24.123	5	5	NULL	Caril	Ferrierio
2	5	lorem quisque ut erat curabitur gravida nisi at...	Armed Robbery	1965-01-12 00:00:00.000	2021-06-20 15:36:24.123	161	5	NULL	Jonathan	Durbin

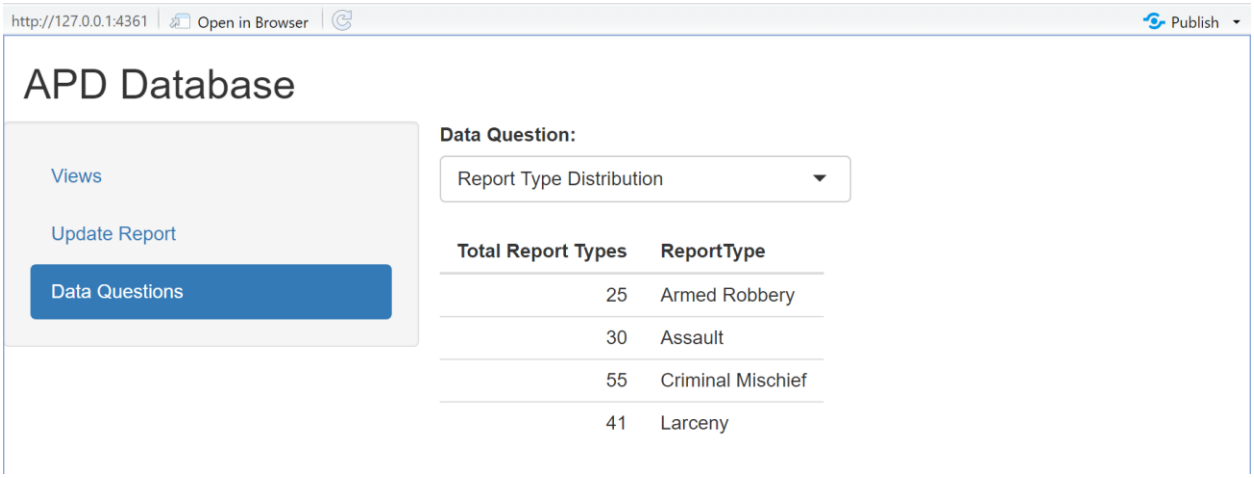
Data Questions Reports:



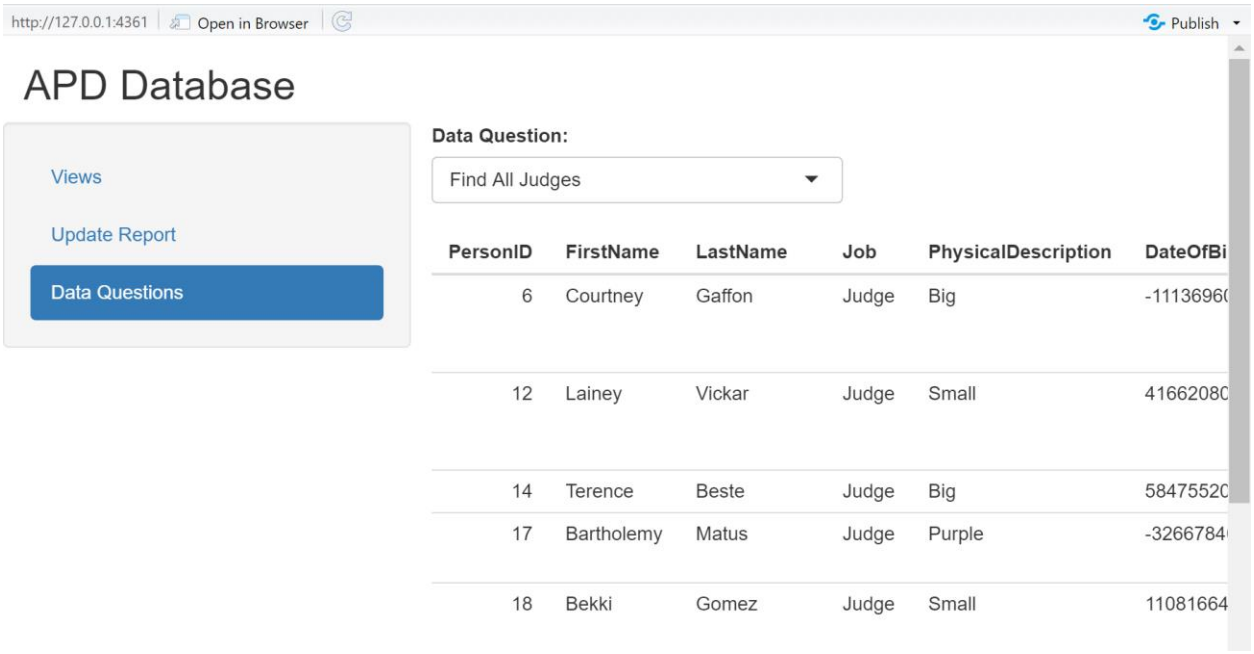




VehicleID	WeaponRegistration	WeaponName	VehicleNumber	VehicleRegist
1	6	AMT Automag V	1	pszz 1110
1	7	AMT Backup	1	pszz 1110
2	42	Berloque pistol	2	kmvj 2271
2	43	Kingdom of Hungary	2	kmvj 2271
3	99	Glisenti Model 1910	3	xefo 0837
3	100	Glock 36	3	xefo 0837
3	101	Glock 37	3	xefo 0837



Total Report Types	ReportType
25	Armed Robbery
30	Assault
55	Criminal Mischief
41	Larceny



R Code

```
library(odbc)
library(DBI)
library(shiny)

ui = fluidPage(
  titlePanel("APD Database"),

  # Check out the views
  navlistPanel(
    tabPanel(
      "Views",
      selectInput(
        "view", "View:",
        c("All Vehicles" = "AllVehicles",
          "All Weapons and Vehicles" = "AllVehiclesWeapons",
          "All Warrants" = "AllWarrants")
      ),
      tableOutput("tbl")
    ),
    tabPanel(
      "Update Report",
      textInput(
        "reportid", "Report ID", placeholder = "Integer"),
      textInput(
        "personresponsible", "Person Responsible", placeholder = "Your ID"),
      textInput(
        "subjectfirstname", "Subject First Name", value = "default"),
```

```
      textInput(
        "subjectlastname", "Subject Last Name", value = "default"),
      textInput(
        "location", "Location", value = "default"),
      textInput(
        "weaponregistration", "Weapon Registration", value = "default"),
      textInput(
        "vehiclenumber", "Vehicle Number", value = "default"),
      actionButton(
        "submitReportUpdate", "Submit Update", placeholder = "Not Required"),
      textOutput("submitReportSuccess")
    ),
    tabPanel(
      "Data Questions",
      selectInput(
        "dq", "Data Question:",
        c("Larceny Report Number" = "larcenyreport",
          "Job Distribution" = "jobdistribution",
          "Vehicles Containing Weapons" = "vehicleweapons",
          "Report Type Distribution" = "reporttypedistribution",
          "Find All Judges" = "alljudges"
          )
      ),
      tableOutput("dqtbl")
    )
  )

)


server <- function(input, output, session) {
  output$tbl <- renderTable({
    conn <-
      dbConnect(
        odbc(),
        Driver = "ODBC Driver 17 for SQL Server",
        Server = "localhost",
        Database = "IST659-Project",
        Trusted_Connection = "Yes"
      )
    on.exit(dbDisconnect(conn), add = TRUE)

    dbGetQuery(
      conn,
      paste0("SELECT * FROM ", input$view, ";")
    )
  })

  observeEvent(
    input$submitReportUpdate,
```

```
    {
      conn <-
        dbConnect(
          odbc(),
          Driver = "ODBC Driver 17 for SQL Server",
          Server = "localhost",
          Database = "IST659-Project",
          Trusted_Connection = "Yes"
        )
      on.exit(dbDisconnect(conn), add = TRUE)

      dbExecute(
        conn,
        paste0("EXEC UpdateReport ",
               input$reportid, ', ',
               input$personresponsible, ', ',
               input$subjectfirstname, ', ',
               input$subjectlastname, ', ',
               input$location, ', ',
               input$weaponregistration, ', ',
               input$vehiclenumber, ';')
      )
    }
  )
  success <- eventReactive(input$submitReportUpdate, { "Executed Procedure!"
})
  output$submitReportSuccess <- renderText({ success() })

  output$dqtbl <- renderTable({
    conn <-
      dbConnect(
        odbc(),
        Driver = "ODBC Driver 17 for SQL Server",
        Server = "localhost",
        Database = "IST659-Project",
        Trusted_Connection = "Yes"
      )
    on.exit(dbDisconnect(conn), add = TRUE)

    data.questions <- c(
      "larcenyreport" = "SELECT COUNT(ReportID) FROM AllReports WHERE ReportT
ype = 'Larceny'",
      "jobdistribution" = "SELECT COUNT(*) AS [Total Jobs], Job FROM Person G
ROUP BY Job",
      "vehicleweapons" = "SELECT * FROM AllVehiclesWeapons",
      "reporttypedistribution" = "SELECT COUNT(*) AS [Total Report Types], Re
portType FROM Report GROUP BY ReportType",
      "alljudges" = "SELECT * FROM Person WHERE Job = 'Judge'"
    )
    dbGetQuery(
```

```
      conn,
      paste0(data.questions[input$dq], ";")
    )
  })
}

# uncomment below:
# shinyApp(ui, server)
```

Reflection

At the start of this project, I held many assumptions about the data structure of a police department. Although I don't know if other departments do things differently than from what the senior clerk I interviewed told me, I can't imagine other departments would need to track very different information. I also didn't realize just how much the possibilities open up when you start to think about what typical operations could be written as stored procedures / functions / views. I know that I only scratched the surface in terms of all the possible programming objects I could have written.

Next time I do this, I'll spend more time in the planning stage. When I was in the planning stage 7/8 weeks ago, I didn't know what to plan for – my inexperience was the primary result of this.

I hope to do this again, many times. This is by far the most fun I've had on a project. I only wish I could work on this more (and get paid to do it!). Next time I do this, I'll think more carefully about what sort of data types I should use, how to handle different logins / users / permissions (I unfortunately didn't do any of this in this project), and how to migrate a database each time I change a column instead of dropping everything / re-creating everything.

After some reflection, I think the job of tracking data for an organization is quite difficult. The question of "do I *really* need to track this information" is particularly hard for me to answer. For example, when building the Vehicles table above, I wondered whether I should keep track of the last time the vehicle was inspected, when its oil was last changed, etc. It was then that I realized that I would have to find a balance between relevant data and data that will never be entered into the system because people are lazy. It's a fine line, at times.

Note: My summary remains at the top of this document (hope that's alright).

Appendix A

```
CREATE DATABASE [IST659-Project]
 CONTAINMENT = NONE
 ON  PRIMARY
( NAME = N'IST659-Project', FILENAME = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\IST659-Project.mdf' , SIZE = 8192KB ,
FILEGROWTH = 65536KB )
 LOG ON
( NAME = N'IST659-Project_log', FILENAME = N'C:\Program Files\Microsoft SQL
Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\IST659-Project_log.ldf' , SIZE = 8192KB ,
FILEGROWTH = 65536KB )
GO
ALTER DATABASE [IST659-Project] SET COMPATIBILITY_LEVEL = 150
GO
ALTER DATABASE [IST659-Project] SET ANSI_NULL_DEFAULT OFF
GO
ALTER DATABASE [IST659-Project] SET ANSI_NULLS OFF
GO
ALTER DATABASE [IST659-Project] SET ANSI_PADDING OFF
GO
ALTER DATABASE [IST659-Project] SET ANSI_WARNINGS OFF
GO
ALTER DATABASE [IST659-Project] SET ARITHABORT OFF
GO
ALTER DATABASE [IST659-Project] SET AUTO_CLOSE OFF
GO
ALTER DATABASE [IST659-Project] SET AUTO_SHRINK OFF
GO
ALTER DATABASE [IST659-Project] SET AUTO_CREATE_STATISTICS ON(INCREMENTAL = OFF)
GO
ALTER DATABASE [IST659-Project] SET AUTO_UPDATE_STATISTICS ON
GO
ALTER DATABASE [IST659-Project] SET CURSOR_CLOSE_ON_COMMIT OFF
GO
ALTER DATABASE [IST659-Project] SET CURSOR_DEFAULT  GLOBAL
GO
ALTER DATABASE [IST659-Project] SET CONCAT_NULL_YIELDS_NULL OFF
GO
ALTER DATABASE [IST659-Project] SET NUMERIC_ROUNDABORT OFF
GO
ALTER DATABASE [IST659-Project] SET QUOTED_IDENTIFIER OFF
GO
ALTER DATABASE [IST659-Project] SET RECURSIVE_TRIGGERS OFF
GO
ALTER DATABASE [IST659-Project] SET  DISABLE_BROKER
GO
ALTER DATABASE [IST659-Project] SET AUTO_UPDATE_STATISTICS_ASYNC OFF
GO
ALTER DATABASE [IST659-Project] SET DATE_CORRELATION_OPTIMIZATION OFF
GO
ALTER DATABASE [IST659-Project] SET PARAMETERIZATION SIMPLE
GO
ALTER DATABASE [IST659-Project] SET READ_COMMITTED_SNAPSHOT OFF
GO
```

```
ALTER DATABASE [IST659-Project] SET  READ_WRITE
GO
ALTER DATABASE [IST659-Project] SET RECOVERY FULL
GO
ALTER DATABASE [IST659-Project] SET  MULTI_USER
GO
ALTER DATABASE [IST659-Project] SET PAGE_VERIFY CHECKSUM
GO
ALTER DATABASE [IST659-Project] SET TARGET_RECOVERY_TIME = 60 SECONDS
GO
ALTER DATABASE [IST659-Project] SET DELAYED_DURABILITY = DISABLED
GO
USE [IST659-Project]
GO
ALTER DATABASE SCOPED CONFIGURATION SET LEGACY_CARDINALITY_ESTIMATION = Off;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET LEGACY_CARDINALITY_ESTIMATION =
Primary;
GO
ALTER DATABASE SCOPED CONFIGURATION SET MAXDOP = 0;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET MAXDOP = PRIMARY;
GO
ALTER DATABASE SCOPED CONFIGURATION SET PARAMETER_SNIFFING = On;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET PARAMETER_SNIFFING = Primary;
GO
ALTER DATABASE SCOPED CONFIGURATION SET QUERY_OPTIMIZER_HOTFIXES = Off;
GO
ALTER DATABASE SCOPED CONFIGURATION FOR SECONDARY SET QUERY_OPTIMIZER_HOTFIXES =
Primary;
GO
USE [IST659-Project]
GO
IF NOT EXISTS (SELECT name FROM sys.filegroups WHERE is_default=1 AND name =
N'PRIMARY') ALTER DATABASE [IST659-Project] MODIFY FILEGROUP [PRIMARY] DEFAULT
GO
```