Gunzip Consulting

**2011**

Gunzip Consulting

Marist College

5/3/2011

Table of Contents

[EXECUTIVE SUMMARY 2](#_Toc292200741)

[ENTITY REALTIONSHIP DIAGRAM 3](#_Toc292200742)

[PERSON TABLE 6](#_Toc292200743)

[STAFF TABLE 8](#_Toc292200744)

[CLIENT TABLE 9](#_Toc292200745)

[EMAIL TABLE 10](#_Toc292200746)

[PHONE TABLE 11](#_Toc292200747)

[COMPANY TABLE 13](#_Toc292200748)

[COMPANY\_INFO TABLE 14](#_Toc292200749)

[PROJECT TABLE 15](#_Toc292200750)

[STAFF\_SPECIALTY TABLE 16](#_Toc292200751)

[SPECIALTY TABLE 18](#_Toc292200752)

[INSERT STATEMENTS 19](#_Toc292200753)

[STORED PROCEDURES 22](#_Toc292200754)

[TRIGGERS 22](#_Toc292200755)

[IMPLEMENTATION DETAILS 22](#_Toc292200756)

[FUTURE ENHANCEMENTS 22](#_Toc292200757)

# EXECUTIVE SUMMARY

Objectives:

As our company, GunZip Consulting, continues to expand we begin to partake in larger more extensive projects. We also begin to increase our client base at an exponential rate with the larger projects. Since not only the amount of projects we have are expanding but the amount of clients per project are also expanding we needed a more efficient way to store information about our company.

We needed to store:

* Information about our staff
* Each of our staff’s area of expertise
* Which of our staff is working with what client
* What project belongs to what client
* Information about our clients
* Information about the clients company
* What client is hosting what project open

Overview:

With the information we need to store we decided against a flat excel sheet to store information because our client base is too large. We decided to develop a relational database on our servers that stores vital information about all our employees, clients, and projects. We made a person who is a super-type which contains basic information about that person and can be a sub-type of either a staff or a client. The staff table links to a staff specialty table which can contain multiple specialties for each of our staff. The client table links to a company table which is tied to a company information table. This contains basic information about the company. The client table links to the project table that the client has open.  The project table also links to the staff member who is working on that project.  We connected all this information so we could query specific information about our company and its clients when needed.

# ENTITY REALTIONSHIP DIAGRAM



**DATA DICTIONARY**

|  |
| --- |
| PERSON |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| PersonID | INT | 0000 - 9999 |  | X |  | Auto Increment |
| FirstName | VARCHAR (20) | A – Z, a - z |  |  |  |  |
| LastName | VARCHAR(30) | A – Z, a - z |  |  |  |  |
| Gender | CHAR(1) | M, F |  |  |  |  |
| AddressLine1 | VARCHAR (100) |  |  |  |  |  |
| AddressLine2 | VARCHAR (100) |  |  |  | X |  |
| AddressLine3 | VARCHAR(100) |  |  |  | X |  |
| City | VARCHAR(30) |  |  |  |  |  |
| State | VARCHAR(20) |  |  |  |  |  |
| Zip | VARCHAR(10) | 1 - 9 |  |  |  |  |

|  |
| --- |
| STAFF |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| PersonID | INT | 0000 - 9999 | X |  |  | Auto Increment |

|  |
| --- |
| STAFF\_SPECIALTY |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| SpecialtyID | INT | 0000 - 9999 | X |  |  | Auto Increment |
| PersonID | INT | 0000 - 9999 | X |  |  | Auto Increment |

|  |
| --- |
| SPECIALTY |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| SpecialtyID | INT | 0000 - 9999 |  | X |  | Auto Increment |
| Specialty | VARCHAR (20) | A – Z, a - z |  |  |  |  |

|  |
| --- |
| CLIENT |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| PersonID | INT | 0000 - 9999 | X |  |  | Auto Increment |
| ContractDate | DATE  |  |  |  |  |  |

|  |
| --- |
| EMAIL |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| EmailID | INT | 0000 - 9999 |  | X |  | Auto Increment |
| PersonID | INT | 0000 - 9999 | X |  |  | Auto Increment |
| EmailAddress | VARCHAR (45) | A – Z, a – z 1 - 9 |  |  |  |  |
| EmailType | VARCHAR(20) | A – Z, a - z |  |  |  |  |

|  |
| --- |
| PHONE |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| PhoneID | INT | 0000 - 9999 |  | X |  | Auto Increment |
| PersonID | INT | 0000 - 9999 | X |  |  | Auto Increment |
| PhoneNumber | VARCHAR (45) | 1 - 9 |  |  |  |  |
| PhoneType | VARCHAR(30) | A – Z, a - z |  |  |  |  |

|  |
| --- |
| PROJECT |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| ProjectID | INT | 0000 - 9999 |  | X |  | Auto Increment |
| ProjectName | VARCHAR (45) | A – Z, a - z |  |  |  |  |
| ProjectDescription | VARCHAR(255) | A – Z, a - z |  |  |  |  |
| StaffID | INT | 0000 - 9999 | X |  |  |  |
| ClientID | INT | 0000 - 9999 | X |  |  |  |

|  |
| --- |
| COMPANY |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| CompanyID | INT | 0000 - 9999 | X |  |  |  |
| PersonID | INT | 0000 - 9999 | X |  |  | Auto Increment |

|  |
| --- |
| COMPANY\_INFO |
| Attribute Name | Data Type | Range | FK | PK | Null | Conditions |
| CompanyID | INT | 0000 - 9999 |  | X |  | Auto Increment |
| CompanyName | VARCHAR (45) | A – Z, a - z |  |  |  |  |

# PERSON TABLE

**PURPOSE:**

 - Identifies anyone in the database and includes their basic information.

**FUNCTIONAL DEPENDENCIES:**

PERSON → PersonID, FirstName, LastName, Gender, AddressLine1, AddressLine2, AddressLine3, City, State, Zip

-- -----------------------------------------------------

-- Table `GUNZIP`.`PERSON`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`PERSON` (

 `PersonID` INT NOT NULL AUTO\_INCREMENT ,

 `FirstName` VARCHAR(20) NOT NULL ,

 `LastName` VARCHAR(30) NOT NULL ,

 `Gender` CHAR NOT NULL ,

 `AddressLine1` VARCHAR(100) NOT NULL ,

 `AddressLine2` VARCHAR(100) NULL ,

 `AddressLine3` VARCHAR(100) NULL ,

 `City` VARCHAR(30) NOT NULL ,

 `State` VARCHAR(20) NOT NULL ,

 `Zip` VARCHAR(10) NOT NULL ,

 PRIMARY KEY (`PersonID`) );

**SAMPLE DATA:**



# STAFF TABLE

**PURPOSE:**

 - Identifies a PERSON as a GUNzip staff member.

**FUNCTIONAL DEPENDENCIES:**

STAFF → *PersonID*

-- -----------------------------------------------------

-- Table `GUNZIP`.`STAFF`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`STAFF` (

 `PersonID` INT NOT NULL AUTO\_INCREMENT ,

 INDEX `fk\_STAFF\_PERSON` (`PersonID` ASC) ,

 CONSTRAINT `fk\_STAFF\_PERSON`

 FOREIGN KEY (`PersonID` )

 REFERENCES `GUNZIP`.`PERSON` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE CASCADE);

**SAMPLE DATA:**



# CLIENT TABLE

**PURPOSE:**

- Identifies a PERSON as a client of GUNzip and lists the date that they became a GUNzip client.

**FUNCTIONAL DEPENDENCIES:**

CLIENT → *PersonID*, ContractedDate

-- -----------------------------------------------------

-- Table `GUNZIP`.`CLIENT`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`CLIENT` (

 `PersonID` INT NOT NULL AUTO\_INCREMENT ,

 `ContractedDate` DATE NOT NULL ,

 INDEX `fk\_CLIENT\_PERSON` (`PersonID` ASC) ,

 CONSTRAINT `fk\_CLIENT\_PERSON`

 FOREIGN KEY (`PersonID` )

 REFERENCES `GUNZIP`.`PERSON` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION);

**SAMPLE DATA:**



# EMAIL TABLE

**PURPOSE:**

- Connects to the PERSON table to allow for anyone in the database to have multiple email addresses (work, personal, etc...).

**FUNCTIONAL DEPENDENCIES:**

EMAIL → EmailID, *PersonID*, EmailAddress, EmailType

-- -----------------------------------------------------

-- Table `GUNZIP`.`EMAIL`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`EMAIL` (

 `EmailID` INT NOT NULL AUTO\_INCREMENT ,

 `PersonID` INT NOT NULL ,

 `EmailAddress` VARCHAR(45) NOT NULL ,

 `EmailType` VARCHAR(20) NOT NULL ,

 PRIMARY KEY (`EmailID`) ,

 INDEX `fk\_EMAIL\_PERSON` (`PersonID` ASC) ,

 CONSTRAINT `fk\_EMAIL\_PERSON`

 FOREIGN KEY (`PersonID` )

 REFERENCES `GUNZIP`.`PERSON` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION);

**SAMPLE DATA:**



# PHONE TABLE

**PURPOSE:**

- Connects to the PERSON table to allow for anyone in the database to have multiple phone numbers (home, cell, work, etc...).

**FUNCTIONAL DEPENDENCIES:**

PHONE → PhoneID, *PersonID*, PhoneNumber, PhoneType

-- -----------------------------------------------------

-- Table `GUNZIP`.`PHONE`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`PHONE` (

 `PhoneID` INT NOT NULL AUTO\_INCREMENT ,

 `PersonID` INT NULL ,

 `PhoneNumber` VARCHAR(45) NOT NULL ,

 `PhoneType` VARCHAR(30) NOT NULL ,

 PRIMARY KEY (`PhoneID`) ,

 INDEX `fk\_PHONE\_PERSON` (`PersonID` ASC) ,

 CONSTRAINT `fk\_PHONE\_PERSON`

 FOREIGN KEY (`PersonID` )

 REFERENCES `GUNZIP`.`PERSON` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION);

**SAMPLE DATA:**



# COMPANY TABLE

**PURPOSE:**

 - Connects the CLIENT table to the COMPANY\_INFO table.

**FUNCTIONAL DEPENDENCIES:**

COMPANY → *CompanyID*, *PersonID*

-- -----------------------------------------------------

-- Table `GUNZIP`.`COMPANY`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`COMPANY` (

 `CompanyID` INT NOT NULL ,

 `PersonID` INT NOT NULL ,

 INDEX `fk\_COMPANY\_CLIENT` (`PersonID` ASC) ,

 INDEX `fk\_COMPANY\_COMANY\_INFO` (`CompanyID` ASC) ,

 CONSTRAINT `fk\_COMPANY\_CLIENT`

 FOREIGN KEY (`PersonID` )

 REFERENCES `GUNZIP`.`CLIENT` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION,

 CONSTRAINT `fk\_COMPANY\_COMANY\_INFO`

 FOREIGN KEY (`CompanyID` )

 REFERENCES `GUNZIP`.`COMPANY\_INFO` (`CompanyID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION);

**SAMPLE DATA:**



# COMPANY\_INFO TABLE

**PURPOSE:**

 - Lists the companies that the individual clients are working for.

**FUNCTIONAL DEPENDENCIES:**

COMPANY\_INFO → *CompanyID*, CompanyName

-- -----------------------------------------------------

-- Table `GUNZIP`.`COMPANY\_INFO`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`COMPANY\_INFO` (

 `CompanyID` INT NOT NULL AUTO\_INCREMENT ,

 `CompanyName` VARCHAR(45) NOT NULL ,

 PRIMARY KEY (`CompanyID`) );

**SAMPLE DATA:**



# PROJECT TABLE

**PURPOSE:**

- Compiles the details of the projects that our staff and clients are working on. Links to the STAFF and CLIENT tables.

**FUNCTIONAL DEPENDENCIES:**

PROJECT → ProjectID, ProjectName, ProjectDescription, *Staff*, *Client*

-- -----------------------------------------------------

-- Table `GUNZIP`.`PROJECT`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`PROJECT` (

 `ProjectID` INT NOT NULL AUTO\_INCREMENT ,

 `ProjectName` VARCHAR(45) NOT NULL ,

 `ProjectDescription` VARCHAR(255) NOT NULL ,

 `Staff` INT NOT NULL ,

 `Client` INT NOT NULL ,

 PRIMARY KEY (`ProjectID`) ,

 INDEX `fk\_PROJECT\_STAFF` (`Staff` ASC) ,

 INDEX `fk\_PROJECT\_CLIENT` (`Client` ASC) ,

 CONSTRAINT `fk\_PROJECT\_STAFF`

 FOREIGN KEY (`Staff` )

 REFERENCES `GUNZIP`.`STAFF` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION,

 CONSTRAINT `fk\_PROJECT\_CLIENT`

 FOREIGN KEY (`Client` )

 REFERENCES `GUNZIP`.`CLIENT` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION);

**SAMPLE DATA:**



# STAFF\_SPECIALTY TABLE

**PURPOSE:**

 - Acts as a connector between the STAFF table and the SPECIALTY table.

**FUNCTIONAL DEPENDENCIES:**

STAFF\_SPECIALTY → *SpecialtyID*, *PersonID*

-- -----------------------------------------------------

-- Table `GUNZIP`.`STAFF\_SPECIALTY`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`STAFF\_SPECIALTY` (

 `SpecialtyID` INT NOT NULL ,

 `PersonID` INT NOT NULL ,

 INDEX `fk\_STAFF\_SPECIALTY\_STAFF` (`PersonID` ASC) ,

 INDEX `fk\_STAFF\_SPECIALTY\_SPECIALTY` (`SpecialtyID` ASC) ,

 CONSTRAINT `fk\_STAFF\_SPECIALTY\_STAFF`

 FOREIGN KEY (`PersonID` )

 REFERENCES `GUNZIP`.`STAFF` (`PersonID` )

 ON DELETE NO ACTION

 ON UPDATE NO ACTION,

 CONSTRAINT `fk\_STAFF\_SPECIALTY\_SPECIALTY`

 FOREIGN KEY (`SpecialtyID` )

 REFERENCES `GUNZIP`.`SPECIALTY` (`SpecialtyID` )

 ON DELETE CASCADE

 ON UPDATE CASCADE);

**SAMPLE DATA:**



# SPECIALTY TABLE

**PURPOSE:**

 - Lists the specialty(s) of each individual staff member.

**FUNCTIONAL DEPENDENCIES:**

SPECIALTY → SpecialtyID, Specialty

-- -----------------------------------------------------

-- Table `GUNZIP`.`SPECIALTY`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `GUNZIP`.`SPECIALTY` (

 `SpecialtyID` INT NOT NULL AUTO\_INCREMENT ,

 `Specialty` VARCHAR(20) NULL ,

 PRIMARY KEY (`SpecialtyID`) );

**SAMPLE DATA:**



# INSERT STATEMENTS

**CLIENT**

INSERT INTO `GUNZIP`.`CLIENT` (`PersonID`, `ContractedDate`) VALUES (6, '2/17/11');
INSERT INTO `GUNZIP`.`CLIENT` (`PersonID`, `ContractedDate`) VALUES (7, '3/1/11');
INSERT INTO `GUNZIP`.`CLIENT` (`PersonID`, `ContractedDate`) VALUES (8, '3/17/11');

 **COMPANY**

INSERT INTO `GUNZIP`.`COMPANY` (`CompanyID`, `PersonID`) VALUES (1, 6);
INSERT INTO `GUNZIP`.`COMPANY` (`CompanyID`, `PersonID`) VALUES (2, 7);
INSERT INTO `GUNZIP`.`COMPANY` (`CompanyID`, `PersonID`) VALUES (3, 8);

 **COMPANY INFO**

INSERT INTO `GUNZIP`.`COMPANY\_INFO` (`CompanyID`, `CompanyName`) VALUES (1, 'Microsoft');
INSERT INTO `GUNZIP`.`COMPANY\_INFO` (`CompanyID`, `CompanyName`) VALUES (2, 'Apple');
INSERT INTO `GUNZIP`.`COMPANY\_INFO` (`CompanyID`, `CompanyName`) VALUES (3, 'Facebook');

**EMAIL**

INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (1, 1, 'Dennis.Lee1@marist.edu', 'School');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (2, 2, 'Andrew.Welfel1@marist.edu', 'School');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (3, 3, 'Christine.Burkhardt1@marist.edu', 'School');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (4, 4, 'Christopher.Barra1@marist.edu', 'School');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (5, 4, 'Christopher.Barra@gmail.com', 'Personal');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (6, 5, 'VJ.Errico1@marist.edu', 'School');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (7, 6, 'BGates@microsoft.com', 'Work');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (8, 7, 'SJobs@apple.com', 'Work');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (9, 8, 'MZuckerberg@facebook.com', 'Work');
INSERT INTO `GUNZIP`.`EMAIL` (`EmailID`, `PersonID`, `EmailAddress`, `EmailType`) VALUES (10, 8, 'Mark.Zuckerberg@gmail.com', 'Personal');
 **PERSON**
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (1, 'Dennis', 'Lee', 'Male', '123 Gunzip Lane', NULL, NULL, 'Poughkeepsie', 'NY', '12601');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (2, 'Andrew', 'Welfel', 'Male', '124 Gunzip Lane', NULL, NULL, 'Poughkeepsie', 'NY', '12601');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (3, 'Christine', 'Burkhardt', 'Female', '125 Gunzip Lane', NULL, NULL, 'Poughkeepsie', 'NY', '12601');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (4, 'Chris', 'Barra', 'Male', '126 Gunzip Lane', NULL, NULL, 'Poighkeepsie', 'NY', '12601');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (5, 'VJ', 'Errico', 'Male', '127 Gunzip Lane', NULL, NULL, 'Poughkeepsie', 'NY', '12601');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (6, 'Bill', 'Gates', 'Male', '1 Microsoft Road', NULL, NULL, 'Seattle', 'WA', '15674');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (7, 'Steve', 'Jobs', 'Male', '1 Apple Road', NULL, NULL, 'Coopertino', 'CA', '12348');
INSERT INTO `GUNZIP`.`PERSON` (`PersonID`, `FirstName`, `LastName`, `Gender`, `AddressLine1`, `AddressLine2`, `AddressLine3`, `City`, `State`, `Zip`) VALUES (8, 'Mark', 'Zuckerberg', 'Male', '1 Facebook Road', NULL, NULL, 'San Diego', 'CA', '17893');
 **PHONE**

INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (1, 1, '181-187-4854', 'HOME');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (2, 1, '574-156-4895', 'CELL');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (3, 2, '541-594-5972', 'HOME');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (4, 2, '231-654-8187', 'CELL');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (5, 3, '897-289-2872', 'HOME');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (6, 4, '345-811-8758', 'HOME');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (7, 4, '487-951-5545', 'CELL');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (8, 5, '675-041-5685', 'HOME');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (9, 5, '978-725-6188', 'CELL');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (10, 6, '821-524-5755', 'CELL');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (11, 7, '681-950-9545', 'WORK');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (12, 8, '223-548-8489', 'WORK');
INSERT INTO `GUNZIP`.`PHONE` (`PhoneID`, `PersonID`, `PhoneNumber`, `PhoneType`) VALUES (13, 8, '256-405-5689', 'CELL');

**PROJECT**

INSERT INTO `GUNZIP`.`PROJECT` (`ProjectID`, `ProjectName`, `ProjectDescription`, `Staff`, `Client`) VALUES (1, 'Microsoft Office Database', 'Microsoft wants us to create a database concerning their Microsoft Office Software', 1, 6);
INSERT INTO `GUNZIP`.`PROJECT` (`ProjectID`, `ProjectName`, `ProjectDescription`, `Staff`, `Client`) VALUES (2, 'Apple Software Assistance', 'Assist in creating Apple operating system software', 2, 7);
INSERT INTO `GUNZIP`.`PROJECT` (`ProjectID`, `ProjectName`, `ProjectDescription`, `Staff`, `Client`) VALUES (3, 'Facebook Website Upgrade', 'Assisting in changing Facebook from PHP to Python.', 3, 8);
 **SPECIALTY**

INSERT INTO `GUNZIP`.`SPECIALTY` (`SpecialtyID`, `Specialty`) VALUES (1, 'Web Design');
INSERT INTO `GUNZIP`.`SPECIALTY` (`SpecialtyID`, `Specialty`) VALUES (2, 'Hardware');
INSERT INTO `GUNZIP`.`SPECIALTY` (`SpecialtyID`, `Specialty`) VALUES (3, 'Software');
INSERT INTO `GUNZIP`.`SPECIALTY` (`SpecialtyID`, `Specialty`) VALUES (4, 'DBA');
 **STAFF**

INSERT INTO `GUNZIP`.`STAFF` (`PersonID`) VALUES (1);
INSERT INTO `GUNZIP`.`STAFF` (`PersonID`) VALUES (2);
INSERT INTO `GUNZIP`.`STAFF` (`PersonID`) VALUES (3);
INSERT INTO `GUNZIP`.`STAFF` (`PersonID`) VALUES (4);
INSERT INTO `GUNZIP`.`STAFF` (`PersonID`) VALUES (5);

**STAFF SPECIALTY**

INSERT INTO `GUNZIP`.`STAFF\_SPECIALTY` (`SpecialtyID`, `PersonID`) VALUES (1, 1);
INSERT INTO `GUNZIP`.`STAFF\_SPECIALTY` (`SpecialtyID`, `PersonID`) VALUES (2, 2);
INSERT INTO `GUNZIP`.`STAFF\_SPECIALTY` (`SpecialtyID`, `PersonID`) VALUES (3, 3);
INSERT INTO `GUNZIP`.`STAFF\_SPECIALTY` (`SpecialtyID`, `PersonID`) VALUES (4, 4);
INSERT INTO `GUNZIP`.`STAFF\_SPECIALTY` (`SpecialtyID`, `PersonID`) VALUES (5, 4);

# STORED PROCEDURES

 There are no stored procedures contained within the first implementation of the data base, but there will be with future improvements since that functionality will be required for those additions.

# TRIGGERS

 As of right now, the only trigger elements within the database are regarding the staff and client tables. When a staff member or client is added the elements are cascaded up to the person table where their information is stored while the PersonID is also contained in each of the relevant tables based on a person’s standing with the company.

# IMPLEMENTATION DETAILS

Our database is like a solar system; the core of it is the PERSON table. With the PERSON table we track all people involved in our company workflow. This makes it easy to associate people with projects, companies and personal information. We also split how we identify people into two distinct groups, Client and Staff. As of right now those are the only two types we need. To do this we have a field in PERSON called PersonType. Using PersonType cascade the personID into the appropriate tables to maintain relationships The two person types again come together though the PROJECT table. Each project has one staff member and one client as the contact for the project. For each Staff member we track their skills and for each client we keep track of their company(s).

 The current state of the database deployment is; the first iteration has been deployed and running. The MySQL instance we used is a base install of MySQL on an Ubuntu 10.04LTS server.

# FUTURE ENHANCEMENTS

Since this is our first implementation of our relational database we still have many features we would like to add in the future. Our first thought for future enhancements is having a receipt made for the client once the project is done and closed out for a point of reference. This receipt table would include the name of the project, our staff member’s information, details about the project, and a price including a list of expenses for the project.  To further track the progress of our active projects, we would like to eventually implement a project progress log.  This would be commentary provided by our staff about the current state of the project for our clients to view. We would also create automatic triggers to notify a client of the next stage of a project informing them the time line of the project and if each stage is behind schedule, on time, or ahead of schedule.  Lastly, we would make a distance calculator to determine the distance between a client and the staff member and its cost effectiveness of a project for our company.