DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 1

Bases de datos

Grupo 4

Integrante	LU	Correo electrónico
Monti, Martin	727/10	Martinmatiasmonti@gmail.com
Goldstein, Brain	27/14	brai.goldstein@gmail.com
Colombo, Ricardo	156/08	ricardogcolombo@gmail.com
Tripodi, Guido	843/10	guido.tripodi@hotmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

Índice

1. Ejercicio 1

1.1. Descripción de problema

Se solicitó la creación de una una base de datos para la Policía de Ciudad Gótica. En dicha base se debió almacenar los datos de los oficiales de la misma (que cuentan con un rango, un número de placa que los identifica, los datos personales y la fecha de ingreso). A su vez, se almacena para los oficiales los cambios de designación.
Un oficial puede tener un sumario relacionado con alguna de sus designaciones. En dicho sumario se almacena la fecha, observación o descripción del sumario, un estado y un resultado. Algunos oficiales pertenecen a Asuntos Internos y son quienes llevan adelante la investigación del sumario.
Cada oficial pertenece a un departamento y éstos contienen un nombre y una descripción. Por otra parte se almacenarán las personas involucradas en incidentes en los que la policía intervenga. De las personas se necesitan saber los datos personales y con que otra persona tienen relaciones. Se debe guardar el historial de domicilios de la persona. Una persona puede participar de un incidente siendo los mismos de diferentes tipos: robo, felonía, violencia doméstica, etc.
Para cada incidente se precisa conocer la ubicación: calle, altura, las calles entre las que ocurrió y barrio. De cada incidente se debe saber además que personas participaron, el rol que jugaron en el incidente (los roles están identificados, como ser: sospechoso, víctima, etc.). Además de las personas se deben almacenar a los oficiales involucrados con el rol que jugaron también.
Las personas se relacionan con otras y debe guardarse esa información con la fecha desde la cual se conoce la relación y que tipo de relación es (amigo, complice, etc).
Los incidentes poseen seguimientos, de los cuales se precisa un número, la fecha, descripción, conclusión un estado (pendiente, cerrado, en proceso).Para los incidentes en proceso, interesa conocer quien está haciendo el seguimiento y la fecha de seguimiento ultima.
La policía tiene identificadas a organizaciones delictivas. Estas organizaciones están codificadas y tienen personas relacionadas a ella. Es importante conocer qué organizaciones están involucradas en incidentes a partir de las personas que las componen.
También es necesario tener almacenados los superheroes que actúan en Ciudad Gótica (Batman, Arrow, etc) de cada uno se conoce su nombre de fantasía, color del disfraz y un conjunto de habilidades. Sólo de algunos se conoce su identidad secreta que sería una persona de la ciudad. Se sabe también que personas tienen contacto con cada superheroe (personas que pueden llamarlos o contactarlos). Algunos superheroes pueden participar de uno o más incidentes. Y además pueden tener uno o varios .ªrchienemigos". Estos .ªrchienemigos"no son más que personas que pueden o no formar parte de alguna organización delictiva.

Ademas, es necesario que nuestra base de datos pueda responder a las siguientes consultas:

- Listado de incidentes en un rango de fechas, mostrando los datos de las personas y policías involucrados con el rol que jugó cada uno en el incidente.
- Dada una organización delictiva, el detalle de incidentes en que participaron las personas que componen dicha organización
- La lista de todos los oficiales con sus rangos, de un departamento dado.
- El ranking de oficiales que participaron en más incidentes
- Los barrios con mayor cantidad de incidentes.
- Todos los oficiales sumariados que participaron de algún incidente.
- Las personas involucradas en incidentes ocurridos en el barrio donde viven

- Los superheroes que tienen una habilidad determinada
- Los superheroes que han participado en algún incidente.
- Listado de todos los incidentes en donde estuvieron involucrados superheroes y fueron causados por los .ªrchienemigos"de los superheroes involucrados.

1.2. Modelo Entidad Relación

En esta primera etapa realizamos un diseño conceptual del problema utilizando la herramienta de Modelo de Entidad Relación. Este lo construimos con la técnica de Diagrama Entidad relación, a continuación presentamos el diagrama obtenido:

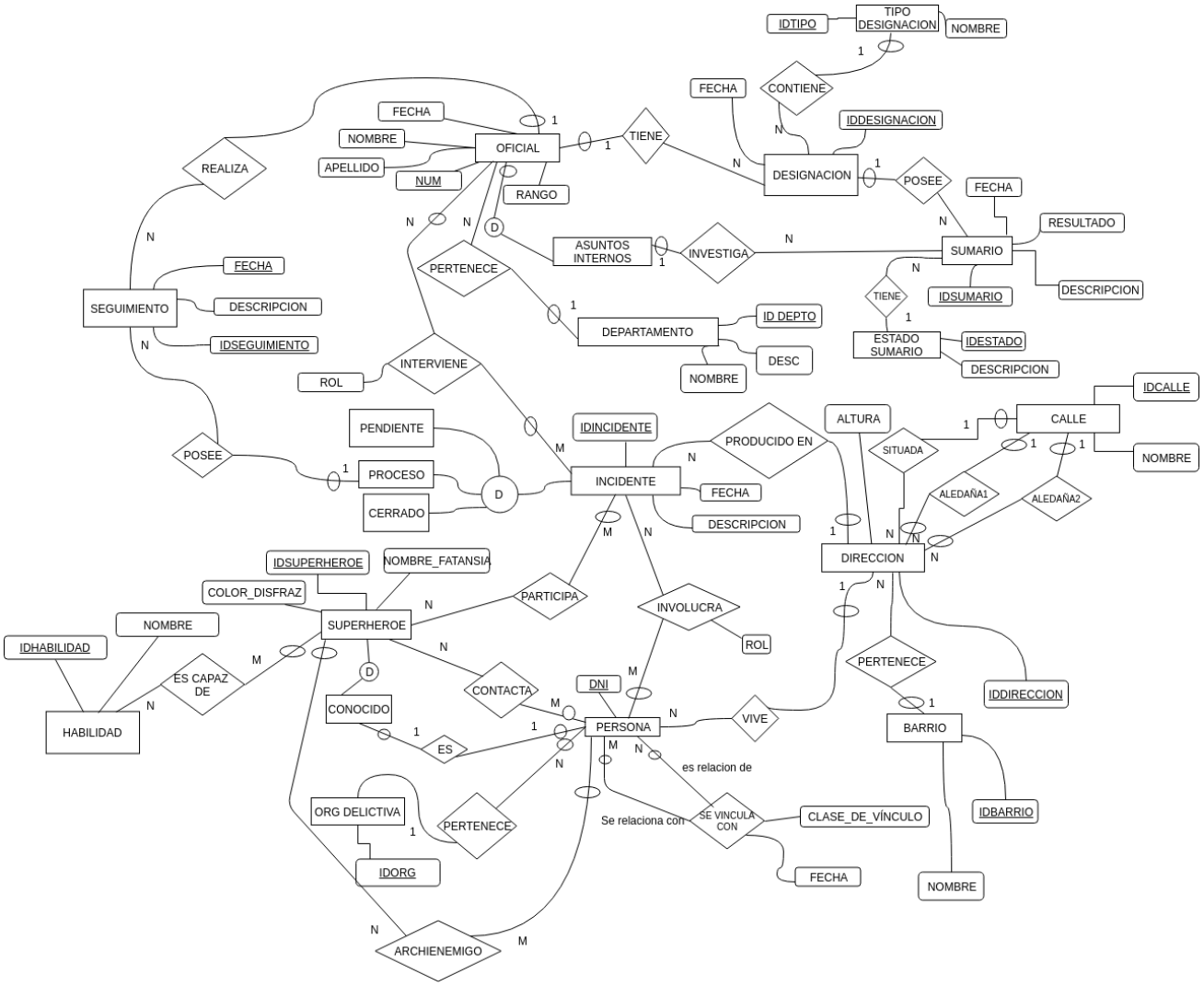

Diagrama Entidad Relación

Para arribar a este modelo, partiendo del enunciado del problema, tuvimos que hacer algunas interpretaciones sobre el mismo, y tomar algunas decisiones donde nos faltó información. Listamos a continuación las decisiones, interpretaciones que respaldan la construcción de nuestro diagrama.

- Cada Seguimiento de un Incidente es confeccionado por un Oficial de Policía, pero contemplamos la posibilidad de que, para un mismo Incidente, distintos Oficiales de Policía carguen Seguimientos (distintos Seguimientos, un mismo Incidente).

- Las calles, por ser elementos identificables, con valores no arbitrarios, y cuyos elementos se repiten en las Direcciones, cobraron en nuestro modelo la suficiente relevancia como para instituirse en una entidad. No así, las alturas de calles en Direcciones.

- Los Roles en los cuales una Persona se Involucra en un Incidente nos parecieron circunscribibles a una lista fija: *Testigo, Víctima, Instigador o Sospechoso*. Sólo trabajamos con esos valores posibles.

- Los Roles en los cuales una Oficial de Policía Interviene en un Incidente nos parecieron circunscribibles a una lista fija:*Designado en manzana, Patrullero, Oficial de civil, Seguridad asignada, Seguridad asignada, Guardia en comisaría*. Sólo trabajamos con esos valores posibles.

- Lo mismo asumimos para la clases de vinculos que podrian llegar a tener dos personas en comun los cuales son: *Ex-compañero de colegio, Socios comerciales, Asociado al mismo club, Afiliacion politica comun, Familiar, Vecino, Indeterminado, Amigo*

Hemos impuestos ciertas restricciones las cuales seran nombradas en el MR

1.3. Modelo Lógico Relacional

En base a nuestro diseño conceptual construido anteriormente, pasamos a un diseño lógico utilizando el Modelo Lógico Relacional que esta determinado por los siguientes esquemas de relación:

Departamento (idDepto, nombre, descripción)

- PK = {idDepto}
- Departamento.idDepto puede no estar en Oficial.departamento

Oficial (númeroPlaca, rango, nombre, apellido, fechaIngreso, idDepartamento)

- PK = {númeroPlaca} FK = {departamento}
- Oficial.idDepartamento debe estar en Departamento.idDepto

Incidente (idIncidente,fecha, descripcion, dirección)

- PK = {idIncidente} FK = {dirección}
- Incidente.dirección debe estar en Dirección.idDirección

Sumario (idSumario,descripción,fecha,resultado, idEstado, idDesignacion, númeroPlaca)

- PK = {idSumario} FK = {idEstado,idDesignacion, númeroPlaca}
- Sumario.estado debe estar en EstadoSumario.idEstado
- Sumario.idDesignacion debe estar en designacion.idDesignacion
- Sumario.númeroPlaca debe estar en Oficial.númeroPlaca
- Sumario.fecha debe ser mayor o igual a designacion.Fecha correspondiente al mismo idDesignacion

EstadoSumario (idEstado,descripción,resultado)

- PK = {idEstado}

Designación (idDesignación, fecha, idTipo, númeroPlaca)

- PK = {idDesignación} FK = {idTipo, númeroPlaca}
- Designación.idTipo debe estar en TipoDesignación.idTipo
- Designación.númeroPlaca debe estar en Oficial.númeroPlaca
- Designacion.fecha debe ser mayor o igual a Oficial.fecha correspondiente al mismo númeroPlaca

TipoDesignación (idTipo, nombre)

- PK = {idTipo}

Seguimiento (idSeguimiento, fecha, descripcion, númeroPlaca, idIncidente)

- PK = {idSeguimiento,fecha } FK = {númeroPlaca,idIncidente }
- Seguimiento.númeroPlaca debe estar en Oficial.númeroPlaca
- Seguimiento.idIncidente debe estar en Proceso.idIncidente
- Seguimiento.fecha debe ser mayor o igual a incidente.Fecha correspondiente al mismo idIncidente
- Seguimiento.fecha debe ser mayor o igual a oficial.fechaIngreso correspondiente al mismo numero-Placa

Direccion (idDireccion, altura, idCalle,idBarrio)

- PK = {idDireccion} FK = {idCalle}
- Direccion.idCalle debe estar en Calle.idCalle
- Direccion.idBarrio debe estar en Barrio.idBarrio

Calle (idCalle, nombre

- PK = {idCalle}

Barrio (idBarrio, Nombre)

- PK = {idBarrio}

Persona (DNI, nombre, apellido, idDireccion,idOrgDelicitiva)

- PK = {DNI} FK = {idDireccion,idOrgDelicitiva}
- Persona.idDireccion debe estar en Direccion.idDireccion
- Persona.idOrgDelicitiva debe estar en OrgDelictiva.idOrgDelicitiva

OrgDelictiva (idOrgDelicitiva, nombre)

- PK = {idOrgDelicitiva}

SuperHeroe (idSuperheroe, nombreFantasia, colorDisfraz)

- PK = {idSuperheroe}

Habilidad (idHabilidad, nombre)

- PK = {idHabilidad}

Conocido (idSuperheroe,DNI)

- PK = {idSuperheroe,DNI } FK = {idSuperheroe,DNI}
- Conocido.idSuperheroe debe estar en SuperHeroe.idSuperheroe
- Conocido.DNI debe estar en SuperHeroe.DNI

Proceso (idIncidente)

- PK = {idIncidente } FK = {idIncidente}
- Proceso.idIncidente debe estar en incidente.idIncidente

Pendiente (idIncidente)

- PK = {idIncidente} FK = {idIncidente}
- Pendiente.idIncidente debe estar en incidente.idIncidente

Cerrado (idIncidente)

- PK = {idIncidente} FK = {idIncidente}
- Cerrado.idIncidente debe estar en incidente.idIncidente

Interviene (númeroPlaca, idIncidente, rol)

- PK = {númeroPlaca,idIncidente } FK = {númeroPlaca,idIncidente}
- Interviene.númeroPlaca debe estar en Oficial.númeroPlaca
- Interviene.idIncidente debe estar en incidente.idIncidente
- Interviene.rol debe ser una de las siguientes opciones: *Designado en manzana—— Patrullero —— Oficial de civil —— Seguridad asignada —— Seguridad asignada —— Guardia en comisaría*

Involucra (DNI, idIncidente, rol)

- PK = {DNI,idIncidente } FK = {DNI,idIncidente}
- Involucra.DNI debe estar en persona.DNI
- Involucra.idIncidente debe estar en incidente.idIncidente
- Involucra.Rol sólo puede ser alguno de los siguientes valores: Testigo, Víctima, Instigador, Sospechoso.

Contacta (DNI, idSuperheroe)

- PK = {DNI,idSuperheroe } FK = {DNI,idSuperheroe}
- Contacta.DNI debe estar en persona.DNI
- Contacta.idSuperheroe debe estar en superHeroe.idSuperheroe

Participa (idIncidente, idSuperheroe)

- PK = {idIncidente,idSuperheroe } FK = {idIncidente,idSuperheroe}
- Participa.idIncidente debe estar en incidente.idIncidente
- Participa.idSuperheroe debe estar en superHeroe.idSuperheroe

Archienemigo (DNI, idSuperheroe)

- PK = {DNI,idSuperheroe } FK = {DNI,idSuperheroe}
- Archienemigo.DNI debe estar en persona.DNI
- Archienemigo.idSuperheroe debe estar en superHeroe.idSuperheroe

1.4. Queries

1.4.1. Listado de incidentes en un rango de fechas, mostrando los datos de las personas y policías involucrados con el rol que jugó cada uno en el incidente

```
delimiter $$
create procedure listadoDeIncidentesPorFecha(in fechaDesde datetime, in fechaHasta datetime)
begin
SELECT
    incidente.Numero AS NumeroIncidente,
    incidente.Descripcion AS Descripcion,
    oficial.num AS NumeroOficial,
    oficial.Nombre AS NombreOficial,
    oficial.Apellido AS ApellidoOficial,
    interviene.Rol AS RolOficial,
    persona.Nombre AS NombrePersona,
    persona.Apellido AS ApellidoPersona,
    involucra.Rol AS RolPersona
FROM
    mydb.Oficial oficial,
    mydb.Incidente incidente,
    mydb.Interviene interviene,
    mydb.Involucra involucra,
    mydb.Persona persona
WHERE
    incidente.Fecha >= fechaDesde
        AND incidente.Fecha <= fechaHasta
        AND incidente.idIncidente = involucra.idIncidente
        AND incidente.idIncidente = interviene.idIncidente
        AND oficial.num = interviene.num
        AND persona.idPersona = involucra.idPersona
GROUP BY incidente.Numero , incidente.Descripcion , oficial.num , oficial.Datos , persona.Nombre
;
            end
$$
```

1.4.2. Dada una organización delictiva, el detalle de incidentes en que participaron las personas que componen dicha organización

```
delimiter $$
create procedure incidentesPorOrganizacion(in organizacion varchar(24))
begin
SELECT
    incidente.Numero AS numeroIncidente,
    incidente.Descripcion AS DescripcionIncidente
FROM
    mydb.OrganizacionDelictiva orgaDelictiva,
    mydb.Persona persona,
    mydb.Involucra involucra,
    mydb.Incidente incidente
WHERE
```

```
    orgaDelictiva.Nombre = organizacion
        AND persona.idOrganizacionDelictiva = orgaDelictiva.idOrganizacionDelictiva
        AND involucra.idPersona = persona.idPersona
        AND incidente.idIncidente = involucra.idIncidente
GROUP BY incidente.Numero , incidente.Descripcion;
end
$$
```

1.4.3. La lista de todos los oficiales con sus rangos, de un departamento dado

```
delimiter $$
create procedure oficialDadoDepartamento(in Nombre varchar(24))
begin
SELECT
    oficial.Nombre AS NombreOficial,
    oficial.Apellido AS ApellidoOficial,
    oficial.Rango AS Rango
FROM
    mydb.Departamento AS departamento,
    mydb.Oficial AS oficial
WHERE
    departamento.Nombre = Nombre
        AND oficial.idDepartamento = departamento.idDepartamento;
end
$$
```

1.4.4. El ranking de oficiales que participaron en más incidentes

```
SELECT
    oficial.num AS NumeroOficial,
    oficial.Nombre AS NombreOficial,
    oficial.Apellido As ApellidoOficial,
    COUNT(*) AS CantidadDeIncidentes
FROM
    mydb.Oficial oficial,
    mydb.Interviene interviene
WHERE
    oficial.num = interviene.num
GROUP BY oficial.num
HAVING COUNT(*) >= ALL (SELECT
        COUNT(*)
    FROM
        mydb.Oficial oficial1,
        mydb.Interviene interviene1
    WHERE
        oficial1.num = interviene1.num
    GROUP BY oficial1.num);
```

1.4.5. Los barrios con mayor cantidad de incidentes

```
SELECT
    barrio.Nombre AS NombreBarrio,
    COUNT(*) AS CantidadDeIncidentes
FROM
    mydb.Barrio barrio,
    mydb.Direccion direccion,
    mydb.Incidente incidente
WHERE
    incidente.idDireccion = direccion.idDireccion
        AND direccion.Barrio_idBarrio = barrio.idBarrio
GROUP BY barrio.idBarrio
HAVING COUNT(*) >= ALL (SELECT
        COUNT(*)
    FROM
        mydb.Barrio barrio1,
        mydb.Direccion direccion1,
        mydb.Incidente incidente1
    WHERE
        incidente1.idDireccion = direccion1.idDireccion
            AND direccion1.Barrio_idBarrio = barrio1.idBarrio
            AND barrio.idBarrio != barrio1.idBarrio
    GROUP BY barrio1.idBarrio);
```

1.4.6. Todos los oficiales sumariados que participaron de algún incidente

```
SELECT
    oficial.Nombre AS NombreOficial,
    oficial.Apellido As ApellidoOficial
FROM
    mydb.Interviene AS interviene,
    mydb.Sumario AS sumario,
    mydb.Oficial AS oficial,
    mydb.Designacion AS designacion
WHERE
    interviene.num = oficial.num
        AND designacion.num = oficial.num
        AND designacion.idDesignacion = sumario.idDesignacion
GROUP BY oficial.Datos;
```

1.4.7. Las personas involucradas en incidentes ocurridos en el barrio donde viven

```
SELECT
    persona.Nombre AS Nombre,
    persona.Apellido AS Apellido,
    barrio.Nombre AS Barrio
FROM
    mydb.Persona persona,
    mydb.Barrio barrio,
```

```
    mydb.Involucra involucra,
    mydb.Incidente incidente,
    mydb.Direccion direccion
WHERE
    persona.idBarrio = barrio.idBarrio
        AND persona.idPersona = involucra.idPersona
        AND incidente.idIncidente = involucra.idIncidente
        AND incidente.idDireccion = direccion.idDireccion
        AND direccion.Barrio_idBarrio = barrio.idBarrio
GROUP BY persona.Nombre , persona.Apellido , barrio.Nombre
;
```

1.4.8. Los superheroes que tienen una habilidad determinada

```
delimiter $
create procedure superheroeConHabilidad(in Habilidad varchar(24))
begin
SELECT superheroe.Nombre FROM mydb.Habilidad AS habilidad, mydb.Superheroe as superheroe
where habilidad.Nombre = Habilidad and superheroe.idSuperheroe = habilidad.Superheroe_idSuperhero
end
$
```

1.4.9. Los superheroes que han participado en algún incidente.

```
SELECT
    Superheroe.Nombre AS Superheroe
FROM
    mydb.Participa AS participa,
    mydb.Superheroe AS superheroe
WHERE
    participa.idSuperheroe = Superheroe.idSuperheroe
GROUP BY Superheroe.Nombre;
```

1.4.10. Listado de todos los incidentes en donde estuvieron involucrados superheroes y fueron causados por los .ªrchienemigos"de los superheroes involucrados

```
SELECT
    incidente.Numero AS NumeroIncidente,
    superheroe.Nombre AS Participa,
    persona.Nombre AS CausadoPor
FROM
    mydb.Superheroe superheroe,
    mydb.Archienemigo archienemigo,
    mydb.Involucra involucra,
    mydb.Participa participa,
    mydb.Incidente incidente,
    mydb.Persona persona
WHERE
    superheroe.idSuperheroe = participa.idSuperheroe
```

```
        AND incidente.idIncidente = participa.idIncidente
        AND archienemigo.idSuperheroe = superheroe.idSuperheroe
        AND archienemigo.idPersona = persona.idPersona
        AND involucra.idPersona = persona.idPersona
        AND involucra.idIncidente = incidente.idIncidente
        AND involucra.Rol LIKE 'ACUSADO'
GROUP BY incidente.Numero , superheroe.Nombre , persona.Nombre
;
```

1.5. Triggers

Para cumplir con algunas restricciones del modelo propuesto implementamos los siguientes triggers

1.5.1. Involucra

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Involucra_BEFORE_INSERT` BEFORE INSERT ON `Involucr
BEGIN
 if(new.Rol != 'Testigo' && new.Rol != 'Víctima' && new.Rol != 'Instigador && new.Rol != 'Sospech
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo ROL debe ser Testigo||Victima||Instigado
end if;
END


CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Involucra_BEFORE_UPDATE` BEFORE UPDATE ON `Involucr
BEGIN
 if(new.Rol != 'Testigo' && new.Rol != 'Víctima' && new.Rol != 'Instigador' && new.Rol != 'Sospec
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo ROL debe ser Testigo||Victima||Instigado
end if;
END
```

1.5.2. Interviene

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Interviene_BEFORE_INSERT` BEFORE INSERT ON `Intervi
BEGIN
 if(new.Rol != 'Designado en manzana' && new.Rol != 'Patrullero' && new.Rol != 'Seguridad asignad
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo ROL es incorrecto';
end if;
END


CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Interviene_BEFORE_UPDATE` BEFORE UPDATE ON `Intervi
BEGIN
 if(new.Rol != 'Designado en manzana' && new.Rol != 'Patrullero' && new.Rol != 'Seguridad asignad
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo ROL es incorrecto';
end if;
END
```

1.5.3. Se relaciona con

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Serelacionacon_BEFORE_INSERT` BEFORE INSERT ON `Ser
BEGIN
 if(new.Tipo != 'Ex-compañero de colegio' && new.Tipo != 'Socios comerciales' && new.Tipo != 'Asc
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo Tipo debe ser Ex-compañero de colegio||S
end if;
END


CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Serelacionacon_BEFORE_UPDATE` BEFORE UPDATE ON `Ser
BEGIN
```

```
if(new.Tipo != 'Ex-compañero de colegio' && new.Tipo != 'Socios comerciales' && new.Tipo != 'Asoc
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo Tipo debe ser Ex-compañero de colegio||S
end if;
END
```

1.5.4. Superheroe

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Superheroe_BEFORE_INSERT` BEFORE INSERT ON `Superhe
BEGIN
if(new.ColorDisfraz != 'Verde' && new.ColorDisfraz != 'Gris' && new.ColorDisfraz != 'Rojo' && new
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo ColorDisfraz no es correcto';
end if;
END
```

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Superheroe_BEFORE_UPDATE` BEFORE UPDATE ON `Superhe
BEGIN
if(new.ColorDisfraz != 'Verde' && new.ColorDisfraz != 'Gris' && new.ColorDisfraz != 'Rojo' && new
 SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Atributo ColorDisfraz no es correcto';
end if;
END
```

1.5.5. Sumario

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Sumario_BEFORE_INSERT` BEFORE INSERT ON `Sumario` F
BEGIN
SET @dateDesignacion = (select fecha From DESIGNACION DES WHERE DES.IDDESIGNACION = NEW.IDDESIGNA
    IF @dateDesignacion > new.fecha
    THEN
              SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'la fecha del sumario no puede ser anter
    END IF;
END
```

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Sumario_BEFORE_UPDATE` BEFORE INSERT ON `Sumario` F
BEGIN
SET @dateDesignacion = (select fecha From DESIGNACION DES WHERE DES.IDDESIGNACION = NEW.IDDESIGNA
    IF @dateDesignacion > new.fecha
    THEN
              SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'la fecha del sumario no puede ser anter
    END IF;
END
```

1.5.6. Seguimiento

```
CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Seguimiento_BEFORE_INSERT` BEFORE INSERT ON `Seguim
BEGIN
        SET @fecha_incidente = (select Fecha From Incidente where incidente.idincidente = new.pro
    IF @fecha_incidente > @new.Fecha
    THEN
```

```
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Seguimiento no puede tener fecha ant
    END IF;
END


CREATE DEFINER = CURRENT_USER TRIGGER `mydb`.`Seguimiento_BEFORE_UPDATE` BEFORE INSERT ON `Seguim
BEGIN
        SET @fecha_incidente = (select Fecha From Incidente where incidente.idincidente = new.pro
    IF @fecha_incidente > @new.Fecha
    THEN
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'El Seguimiento no puede tener fecha ant
    END IF;
END
```

2. Conclusiones

A lo largo del diseño e implementación del modelo de datos nos encontramos con estas etapas como las más significativas que nos genero contratiempos.

- Modelización de entidades
- Refinamientos en el MER y MR
- Implementación de queries

Modelización de entidades

A la hora de realizar el modelo entidad relación, de forma inicial se planteo la existencia de una entidad débil Habilidad de Superheroe, al realizar esto nos dimos cuenta luego que mas de un superheroe podria llegar a tener la misma habilidad, por lo tanto le quitamos la condicion de debil dejando una relacion del tipo M:N con superheroe.

Refinamientos en el MER y MR

El pasaje del MER al MR fue sistemático, pero a medida que se conocían requerimientos adicionales del problema se produjeron varios refinamientos sobre el MER de forma iterativa, lo que subsecuentemente también ocasiono cambios en el MR. Sumado a lo enunciado anteriormente, sufrimos contratiempos a la hora de refinar la entidad Direccion, ya que notamos que Calle se solia repetir lo que nos llevo a realizar una modificación y pasar de Parametro Calle a entidad Calle, la cual esta relacionada (1:N) directamente con Direccion. También nos encontramos con ciertas restricciones que se habían pasado por alto a la hora de diseñar. (Restringimos parametros del tipo Rol, Color, y Vinculo por ej)

Implementación de queries

Las queries relacionadas con la obtención de los máximos representaron las de mayor dificultad a la hora de programar. Errores de datos duplicados por errores en los GROUP BY o la falta de HAVING en la query fueron algunos de nuestros errores más comunes.

3. Aclaraciones para correr las implementaciones

Para montar el proyecto se deberá tener instalado el motor de base de datos Mysql Server 5.7 Server compatibles con MySQL en su versión más nueva, trabajamos con la interfaz MySQL Workbench versión 6.3.

Para crear la base de datos se deberá correr **tp1.sql** el cual contiene el DDL. Tanto las queries de los distintos ejercicios como los store procedures se encuentran en la carpeta ”queries”. Se generaron inserts de prueba para el testeo de la base: los mismos se encuentran en el archivo **mockDataInserts.sql** en la carpeta raiz del proyecto.