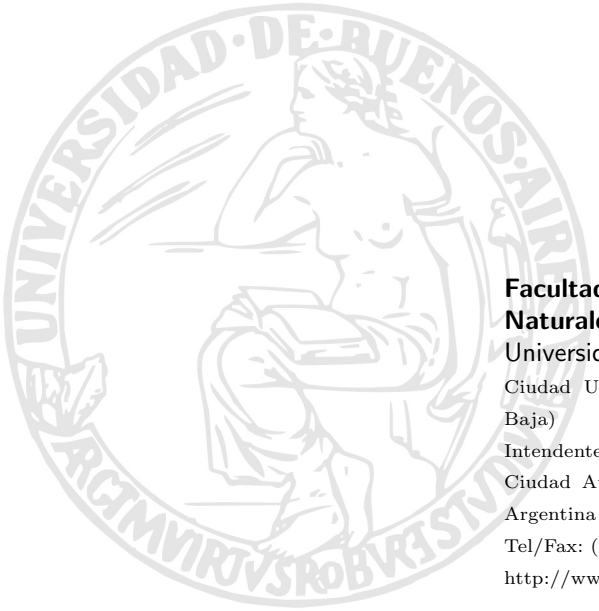
DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Trabajo Práctico 1

4 de febrero de 2019 — Bases de Datos



Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

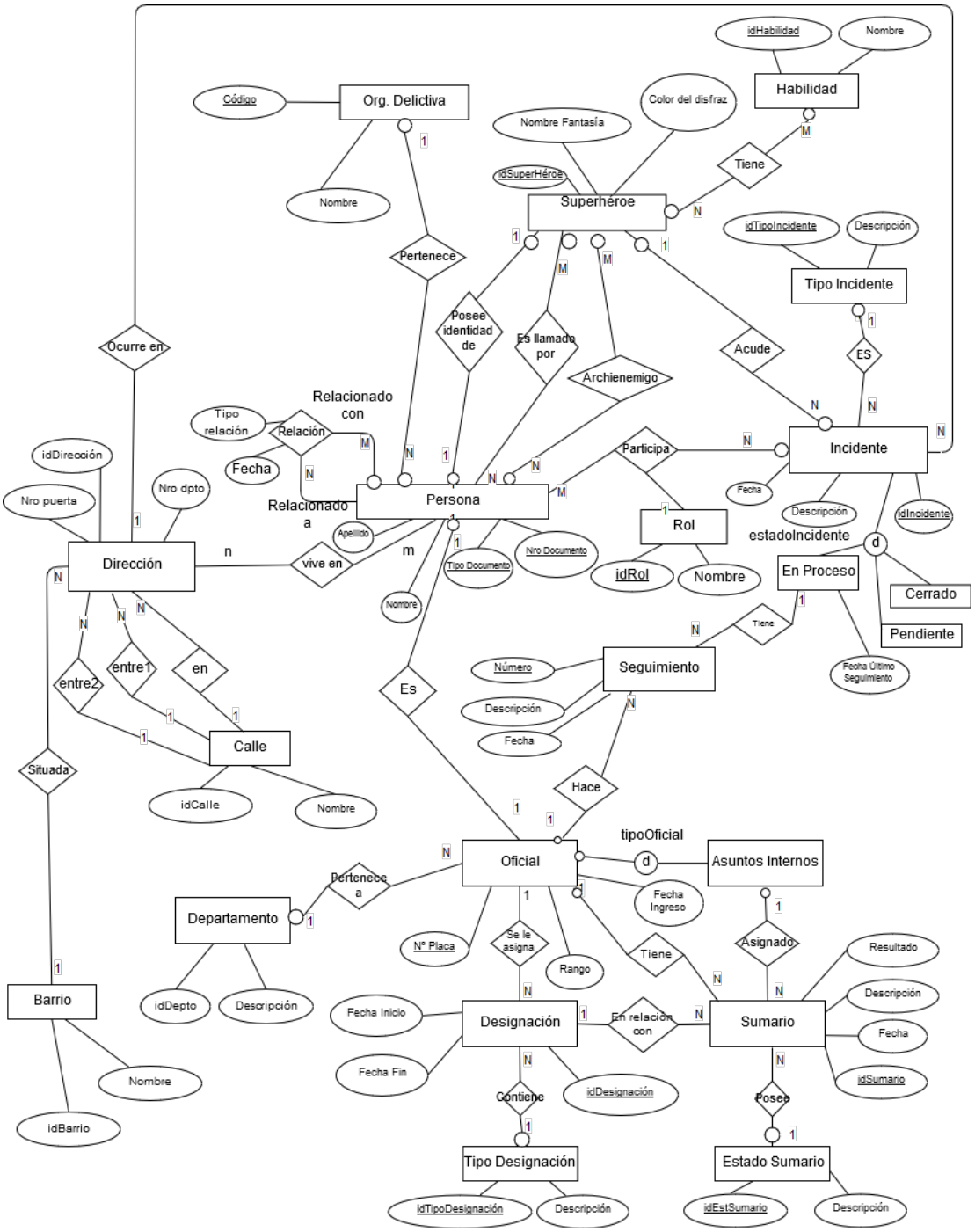
1. Introducción

En el presente trabajo se realiza un modelo de relación de la **Policía de Ciudad Gótica**, para el cual se utilizan herramientas de modelado y, desde un punto de vista lógico, se confecciona un diagrama acorde a las necesidades planteadas.

Las tareas anteriormente mencionadas permitieron la creación de una base de datos, que resulta apropiada para la **Policía de Ciudad Gótica**. La misma es implementada en el motor de base de datos **MySQL**.

2. Modelo de Entidad Relación y Modelo Relacional Derivado

2.1. Modelo de Entidad Relación (DER)



2.2. Modelo Relacional Derivado (MR)

EstadoSumario(idEstSumario, Descripcion)

TipoDesignacion(idTipoDesignacion, Descripcion)

Designacion(idDesignacion, FechaInicio, FechaFin, idTipoDesignacion, NroPlaca)

Sumario(idSumario, Resultado, Descripcion, Fecha,idEstSumario, idOficial, idOficialAsignado, idDesignacion)

Departamento(idDepto, Descripcion)

Oficial(NroPlaca, Rango, FechaIngreso, TipoDocumento, NroDocumento, idDepto, TipoOficial)

OficialAsuntosInternos(NroPlaca)

OrgDelictiva(Codigo, Nombre)

Persona(TipoDocumento, NroDocumento, Nombre, Apellido, Codigo)

Relacion(TipoDocumentoRelacionadoCon, NroDocumentoRelacionadoCon, TipoDocumentoRelacionadoA, NroDocumentoRelacionadoA, TipoRelacion, Fecha)

ViveEn (TipoDocumento, NroDocumento, idDireccion)

Barrio (idBarrio, Nombre)

Calle (idCalle, Nombre)

Direccion (idDireccion, idCalleEntre1, idCalleEntre2, idCalleEn, idBarrio, NroPuerta, NroDpto)

Habilidad(idHabilidad, Nombre)

Superheroe(idSuperHeroe, NombreFantasia, ColorDelDisfraz, TipoDocumento, NroDocumento)

Tiene(idSuperHeroe, idHabilidad)

Archienemigo(TipoDocumento, NroDocumento, idSuperHeroe)

EsLlamadoPor(TipoDocumento, NroDocumento, idSuperHeroe)

TipoIncidente(idTipoIncidente, Descripcion)

Incidente(idIncidente, Descripcion, idTipoIncidente, idSuperHeroe, idDireccion, estadoIncidente, fecha)

IncidenteCerrado (idIncidente)

IncidentePendiente (idIncidente)

IncidenteEnProceso(idIncidente, FechaUltimoSeguimiento)

Seguimiento(Numero, Descripcion, Fecha, idIncidente, NroPlaca)

Rol(idRol, Nombre)

Participa(TipoDocumento, NroDocumento, idIncidente, idRol)

3. Desarrollo

3.1. Supuestos Asumidos

- El tipo de documento es un enumerado. En los datos de prueba se utilizo el 1 para hacer referencia al Documento Único.
- El tipo de relación también se asumió como enumerado. Se utilizaron:
  - 1 - Familiar
  - 2 - Amigo
  - 3 - Conocido
  - 4 - Compañero de trabajo.
- Un incidente siempre tiene algún estado al ser creado.
- Un incidente tiene que estar en un único estado, por lo cual, cada vez que un incidente cambia de estado es necesario asegurarse de que esta restricción sea respetada.
- El rol es "Policia", solo lo pueden ser asignado a oficiales.

3.2. Diseño de Base de Datos para MySQL

3.2.1. Tablas

```
create table EstadoSumario (
    idEstSumario integer not null auto_increment,
    Descripción varchar(50) not null,
    primary key (idEstSumario)
);

create table TipoDesignacion(
    idTipoDesignacion integer not null auto_increment,
    Descripcion varchar(50) not null,
    primary key (idTipoDesignacion)
```

```
);

create table Designacion(
    idDesignacion integer not null auto_increment,
    FechaInicio datetime not null,
    FechaFin datetime,
    idTipoDesignacion integer not null,
    NroPlaca integer not null,
    primary key (idDesignacion)
);

create table OrgDelictiva(
    Codigo integer not null auto_increment,
    Nombre varchar(255) not null,
    primary key (Codigo)
);

create table Persona(
    TipoDocumento integer not null,
    NroDocumento integer not null,
    Nombre varchar(100) not null,
    Apellido varchar(100) not null,
    Codigo integer,
    primary key (TipoDocumento, NroDocumento)
);

create table Departamento(
    idDepto integer not null auto_increment,
    Descripcion varchar(50),
    primary key (idDepto)
);

create table Oficial(
    NroPlaca integer not null auto_increment,
    Rango varchar(50) not null,
    FechaIngreso datetime not null,
    TipoDocumento integer not null,
    NroDocumento integer not null,
    idDepto integer not null,
    TipoOficial varchar(50),
    primary key (NroPlaca)
);

create table OficialAsuntosInternos (
    NroPlaca integer not null,
    primary key (NroPlaca)
);

create table Sumario(
    idSumario integer not null auto_increment,
```

```
    Resultado varchar(50),
    Descripcion varchar(255),
    Fecha datetime not null,
    idEstSumario integer not null,
    idOficial integer not null,
    idOficialAsignado integer not null,
    idDesignacion integer not null,
    primary key (idSumario)
);

create table Relacion(
    TipoDocumentoRelacionadoCon integer not null,
    NroDocumentoRelacionadoCon integer not null,
    TipoDocumentoRelacionadoA integer not null,
    NroDocumentoRelacionadoA integer not null,
    TipoRelacion integer not null,
    Fecha datetime not null,
    primary key (TipoDocumentoRelacionadoCon,NroDocumentoRelacionadoCon,
                TipoDocumentoRelacionadoA,NroDocumentoRelacionadoA)
);

create table Barrio (
 idBarrio integer not null auto_increment,
 Nombre varchar(50) not null,
 primary key (idBarrio)
);

create table Calle (
    idCalle integer not null auto_increment,
    Nombre varchar(50),
    primary key (idCalle)
);

create table Direccion (
    idDireccion integer not null auto_increment,
    idCalleEntre1 integer not null,
    idCalleEntre2 integer not null,
    idCalleEn integer not null,
    idBarrio integer not null,
    NroPuerta integer,
    NroDpto varchar(20),
    primary key (idDireccion)
);

create table ViveEn (
    TipoDocumento integer not null,
    NroDocumento integer not null,
    idDireccion integer not null,
    primary key (TipoDocumento,NroDocumento,idDireccion)
);
```

```
create table Habilidad(
    idHabilidad integer not null auto_increment,
    Nombre varchar(50) not null,
    primary key (idHabilidad)
);

create table Superheroe(
    idSuperHeroe integer not null auto_increment,
    NombreFantasia varchar(50) not null,
    ColorDelDisfraz varchar(50) not null,
    TipoDocumento integer,
    NroDocumento integer,
    primary key (idSuperHeroe)
);

create table Tiene(
 idSuperHeroe integer not null,
 idHabilidad integer not null,
    primary key (idSuperHeroe, idHabilidad)
);

create table Archienemigo (
    TipoDocumento integer not null,
    NroDocumento integer not null,
    idSuperHeroe integer not null,
    primary key (TipoDocumento, NroDocumento, idSuperHeroe)
);

create table EsLlamadoPor(
    TipoDocumento integer not null,
    NroDocumento integer not null,
    idSuperHeroe integer not null,
    primary key (TipoDocumento, NroDocumento, idSuperHeroe)
);

create table TipoIncidente(
    idTipoIncidente integer not null auto_increment,
    Descripcion varchar(100) not null,
    primary key (idTipoIncidente)
);

create table Incidente(
    idIncidente integer not null auto_increment,
    Descripcion varchar(255) not null,
    idTipoIncidente integer not null,
    idSuperHeroe integer,
    idDireccion integer not null,
    estadoIncidente integer,
    fecha dateTime,
```

```
    primary key (idIncidente)
);

create table IncidenteCerrado (
    idIncidente integer not null,
    primary key (idIncidente)
);

create table IncidentePendiente (
    idIncidente integer not null,
    primary key (idIncidente)
);

create table IncidenteEnProceso(
    idIncidente integer not null,
    FechaUltimoSeguimiento datetime not null,
    primary key (idIncidente)
);

create table Seguimiento(
    Numero integer not null auto_increment,
    Descripcion varchar(255) not null,
    Fecha datetime not null,
    idIncidente integer not null,
    NroPlaca integer not null,
    primary key (Numero)
);

create table Rol(
    idRol integer not null auto_increment,
    Nombre varchar(50) not null,
    primary key (idRol)
);

create table Participa(
    TipoDocumento integer not null,
    NroDocumento integer not null,
    idIncidente integer not null,
    idRol  integer not null,
    primary key (TipoDocumento, NroDocumento, idIncidente)
);
```

3.2.2. Foreign Keys

```
ALTER TABLE Designacion
ADD foreign key FK_TipoDesignacion(idTipoDesignacion)
REFERENCES TipoDesignacion(idTipoDesignacion);

ALTER TABLE Designacion
```

```
ADD foreign key FK_DesignacionOficial(NroPlaca)
REFERENCES Oficial(NroPlaca);

ALTER TABLE Persona
ADD foreign key FK_PersonaOrgDelictiva(Codigo)
REFERENCES OrgDelictiva(Codigo);

ALTER TABLE Oficial
ADD foreign key FK_OficialDepartamento(idDepto)
REFERENCES Departamento(idDepto);

ALTER TABLE Oficial
ADD foreign key FK_OficialPersona(TipoDocumento,NroDocumento)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE OficialAsuntosInternos
ADD foreign key FK_OficialAsuntosInternos(NroPlaca)
REFERENCES Oficial(NroPlaca);

ALTER TABLE Sumario
ADD foreign key FK_EstadoSumario(idEstSumario)
REFERENCES EstadoSumario(idEstSumario);

ALTER TABLE Sumario
ADD foreign key FK_SumarioOficial(idOficial)
REFERENCES Oficial(NroPlaca);

ALTER TABLE Sumario
ADD foreign key FK_SumarioOficialAsuntosInternos(
idOficialAsignado)
REFERENCES OficialAsuntosInternos(NroPlaca);

ALTER TABLE Sumario
ADD foreign key FK_SumarioDesignacion(idDesignacion)
REFERENCES Designacion(idDesignacion);

ALTER TABLE Relacion
ADD foreign key FK_PersonaRelacionadoCon(
TipoDocumentoRelacionadoCon,NroDocumentoRelacionadoCon)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE Relacion
ADD foreign key FK_PersonaRelacionadoA(
TipoDocumentoRelacionadoA,NroDocumentoRelacionadoA)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE Direccion
ADD foreign key FK_DireccionEntreCalle1(idCalleEntre1)
REFERENCES Calle(idCalle);
```

```
ALTER TABLE Direccion
ADD foreign key FK_DireccionEntreCalle2(idCalleEntre2)
REFERENCES Calle(idCalle);

ALTER TABLE Direccion
ADD foreign key FK_DireccionCalle(idCalleEn)
REFERENCES Calle(idCalle);

ALTER TABLE Direccion
ADD foreign key FK_DireccionBarrio(idBarrio)
REFERENCES Barrio(idBarrio);

ALTER TABLE ViveEn
ADD foreign key FK_PersonaViveEn(TipoDocumento,NroDocumento)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE ViveEn
ADD foreign key FK_ViveEnDireccion(idDireccion)
REFERENCES Direccion(idDireccion);

ALTER TABLE Superheroe
ADD foreign key FK_PersonaSuperheroe(TipoDocumento,
NroDocumento)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE Tiene
ADD foreign key FK_SuperHeroeTiene(idSuperHeroe)
REFERENCES Superheroe(idSuperHeroe);

ALTER TABLE Tiene
ADD foreign key FK_HabilidadTiene(idHabilidad)
REFERENCES Habilidad(idHabilidad);

ALTER TABLE Archienemigo
ADD foreign key FK_PersonaArchienemigo(TipoDocumento,
NroDocumento)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE Archienemigo
ADD foreign key FK_SuperHeroeArchienemigo(idSuperHeroe)
REFERENCES Superheroe(idSuperHeroe);

ALTER TABLE EsLlamadoPor
ADD foreign key FK_EsLlamadoPorPersona(TipoDocumento,
NroDocumento)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE EsLlamadoPor
ADD foreign key FK_SuperHeroeEsLlamadoPor(idSuperHeroe)
REFERENCES Superheroe(idSuperHeroe);
```

```
ALTER TABLE Incidente
ADD foreign key FK_IncidenteTipoIncidente(idTipoIncidente)
REFERENCES TipoIncidente(idTipoIncidente);

ALTER TABLE Incidente
ADD foreign key FK_IncidenteSuperHeroe(idSuperHeroe)
REFERENCES Superheroe(idSuperHeroe);

ALTER TABLE Incidente
ADD foreign key FK_IncidenteDireccion(idDireccion)
REFERENCES Direccion(idDireccion);

ALTER TABLE IncidenteCerrado
ADD foreign key FK_IncidenteCerrado(idIncidente)
REFERENCES Incidente(idIncidente);

ALTER TABLE IncidentePendiente
ADD foreign key FK_IncidentePendiente(idIncidente)
REFERENCES Incidente(idIncidente);

ALTER TABLE IncidenteEnProceso
ADD foreign key FK_IncidenteEnProceso(idIncidente)
REFERENCES Incidente(idIncidente);

ALTER TABLE Seguimiento
ADD foreign key FK_SeguimientoIncidenteEnProceso(idIncidente)
REFERENCES IncidenteEnProceso(idIncidente);

ALTER TABLE Seguimiento
ADD foreign key FK_SeguimientoOficial(NroPlaca)
REFERENCES Oficial(NroPlaca);

ALTER TABLE Participa
ADD foreign key FK_PersonaParticipa(TipoDocumento,
NroDocumento)
REFERENCES Persona(TipoDocumento,NroDocumento);

ALTER TABLE Participa
ADD foreign key FK_ParticipaIncidente(idIncidente)
REFERENCES Incidente(idIncidente);

ALTER TABLE Participa
ADD foreign key FK_RolParticipa(idRol)
REFERENCES Rol(idRol);
```

3.2.3. Triggers

```
DELIMITER $$
CREATE TRIGGER tg_insert_inc_pendiente BEFORE INSERT
```

```
ON incidentePendiente
FOR EACH ROW
BEGIN
    DELETE FROM incidenteenproceso WHERE idIncidente = new.idIncidente;
    DELETE FROM incidenteCerrado WHERE idIncidente = new.idIncidente;
END$$
DELIMITER ;

DELIMITER $$
CREATE TRIGGER tg_insert_inc_enporceso BEFORE INSERT
ON incidenteenproceso
FOR EACH ROW
BEGIN
    DELETE FROM incidentePendiente WHERE idIncidente = new.idIncidente;
    DELETE FROM incidenteCerrado WHERE idIncidente = new.idIncidente;
END$$
DELIMITER ;

DELIMITER $$
CREATE TRIGGER tg_insert_inc_cerrado BEFORE INSERT
ON incidenteCerrado
FOR EACH ROW
BEGIN
    DELETE FROM incidentePendiente WHERE idIncidente = new.idIncidente;
    DELETE FROM incidenteenproceso WHERE idIncidente = new.idIncidente;
END$$
DELIMITER ;
```

3.3. Consultas SQL

- 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.

```
SELECT i.idIncidente, i.Descripcion , t.Descripcion,
(CASE p.TipoDocumento WHEN 1 THEN 'DU' ELSE 'N/C' END) as Documento,
p.NroDocumento, p.Nombre, p.Apellido,r.nombre
FROM Incidente i
INNER JOIN TipoIncidente t on t.idTipoIncidente = i.idTipoIncidente
INNER JOIN Participa par on par.idIncidente = i.idIncidente
INNER JOIN Persona p on p.tipoDocumento = par.tipoDocumento
            and p.nroDocumento = par.nroDocumento
INNER JOIN Rol r on r.idrol = par.idRol
WHERE i.fecha >=  '2018-05-25' and i.fecha <= '2018-10-25';
```

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

```
SELECT o.Nombre,
```

```
(CASE p.TipoDocumento WHEN 1 THEN 'DU' ELSE 'N/C' END) as Documento,
p.NroDocumento, p.Nombre, p.Apellido, i.idIncidente, i.Descripcion
FROM OrgDelictiva o
INNER JOIN Persona p on p.codigo = o.codigo
INNER JOIN Participa par on p.tipoDocumento = par.tipoDocumento
            and p.nroDocumento = par.nroDocumento
INNER JOIN Incidente i on i.idIncidente = par.idIncidente;
```

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

```
SELECT (CASE p.TipoDocumento WHEN 1 THEN 'DU' ELSE 'N/C' END) as Documento,
p.NroDocumento, p.Nombre, p.Apellido, o.rango, d.descripcion
FROM Oficial o
INNER JOIN Persona p on p.tipoDocumento = o.tipoDocumento
            and p.nroDocumento = o.nroDocumento
INNER JOIN Departamento d on o.idDepto = d.idDepto
WHERE d.idDepto = 3;
```

- El ranking de oficiales que participaron en más incidentes.

```
SELECT COUNT(i.idIncidente) as 'Cant Incidentes', o.NroPlaca
    , p.Nombre, p.Apellido, o.rango
FROM Oficial o
INNER JOIN Persona p on p.tipoDocumento = o.tipoDocumento
    and p.nroDocumento = o.nroDocumento
INNER JOIN Participa par on p.tipoDocumento = par.
    tipoDocumento and p.nroDocumento = par.nroDocumento
INNER JOIN Incidente i on i.idIncidente = par.idIncidente
GROUP BY o.NroPlaca, p.Nombre, p.Apellido, o.rango
HAVING COUNT(i.idIncidente) = (SELECT MAX(y.cant)

            FROM (SELECT COUNT(idIncidente) cant FROM
    Participa ph

            INNER JOIN Oficial oh on oh.tipoDocumento = ph.
    tipoDocumento and oh.nroDocumento = ph.nroDocumento

            GROUP BY  ph.nroDocumento , ph.tipoDocumento) y);
```

- Los barrios con mayor cantidad de incidentes.

```
SELECT COUNT(b.Nombre) as 'Cant Incidentes', b.Nombre
FROM Barrio b
INNER JOIN Direccion d on b.idBarrio = d.idBarrio
INNER JOIN Incidente i on d.idDireccion = i.idDireccion
GROUP BY b.Nombre
HAVING  COUNT(i.idIncidente) = (SELECT MAX(y.cant)
FROM (SELECT count(idBarrio) cant FROM Incidente ih
INNER JOIN Direccion dh on ih.idDireccion = dh.idDireccion
GROUP BY dh.idBarrio) y);
```

- Todos los oficiales sumariados que participaron de algún incidente.

```
SELECT DISTINCT o.NroPlaca, p.Nombre, p.Apellido, o.rango FROM Sumario s
INNER JOIN Oficial o on o.nroPlaca = s.idOficial
INNER JOIN Persona p on p.tipoDocumento = o.tipoDocumento
                and p.nroDocumento = o.nroDocumento
INNER JOIN Participa par on p.tipoDocumento = par.tipoDocumento
                and p.nroDocumento = par.nroDocumento;
```

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

```
SELECT (CASE p.TipoDocumento WHEN 1 THEN 'DU' ELSE 'N/C' END) as Documento,
p.NroDocumento, p.Nombre, p.Apellido,bi.Nombre
FROM Persona p
INNER JOIN Participa par on p.tipoDocumento = par.tipoDocumento
                and p.nroDocumento = par.nroDocumento
INNER JOIN Incidente i on i.idIncidente = par.idIncidente
INNER JOIN Direccion di on di.idDireccion = i.idDireccion
INNER JOIN Barrio bi on bi.idBarrio = di.idBarrio
INNER JOIN ViveEn v on p.tipoDocumento = v.tipoDocumento
                and p.nroDocumento = v.nroDocumento
INNER JOIN Direccion dp on dp.idDireccion = v.idDireccion
WHERE bi.idBarrio = dp.idBarrio;
```

- Los superheroes que tienen una habilidad determinada.

```
SELECT s.NombreFantasia, h.Nombre
FROM Superheroe s
INNER JOIN Tiene t on s.idSuperHeroe = t.idSuperHeroe
INNER JOIN Habilidad h on h.idHabilidad = t.idHabilidad
WHERE h.idHabilidad = 4;
```

- Los superheroes que han participado en algún incidente.

```
SELECT NombreFantasia
FROM Superheroe
WHERE idSuperHeroe in (SELECT idSuperHeroe FROM Incidente);
```

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

```
SELECT i.idIncidente, i.Descripcion, s.NombreFantasia, p.nombre,
p.apellido
FROM Incidente i
INNER JOIN Superheroe s on s.idSuperHeroe = i.idSuperHeroe
INNER JOIN Participa par on par.idIncidente = i.idIncidente
INNER JOIN Persona p on p.tipoDocumento = par.tipoDocumento
                and p.nroDocumento = par.nroDocumento
INNER JOIN Archienemigo a on p.tipoDocumento = a.tipoDocumento
                and p.nroDocumento = a.nroDocumento
                and a.idSuperHeroe = s.idSuperHeroe;
```

4. Conclusiones

Dada la complejidad del escenario planteado por el enunciado del presente trabajo, se puede observar como las herramientas DER (Diagrama de Entidad-Relación) y MR (Modelo Relacional) permiten simplificar el entendimiento del problema y su propio dominio, al trabajar en cada etapa en un aspecto particular del problema.

El DER permite visualizar de manera directa las relaciones entre las distintas entidades y por lo tanto, ayuda a la modelización del problema posponiendo todos los detalles implementativos (que si se considerarían en esta etapa, complejizarían la solución). El MR derivado permite volcar todos esos detalles de relaciones en algo más aproximado a un diseño físico, teniendo en cuenta como serán las relaciones formadas en una base de datos relacional estándar.

Combinando la solución de estas dos herramientas, se obtiene un diseño físico de manera inmediata que es fiel al dominio del problema que se intentaba resolver, manteniendo todas sus propiedades.

Dado lo mencionado previamente, consideramos que la aplicación de estas herramientas en cualquier diseño de base de datos es esencial para poder resolver cualquier modelización respecto a un problema dado.