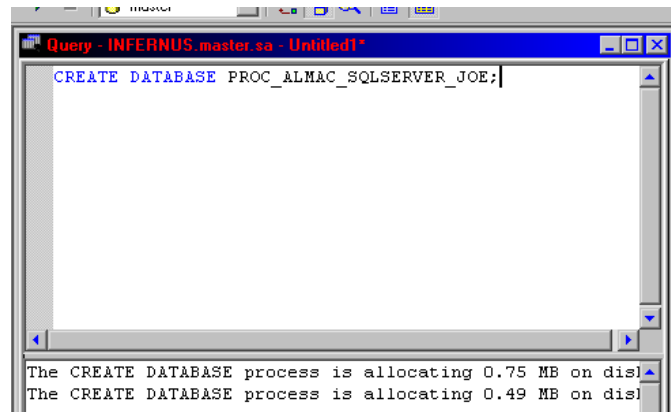
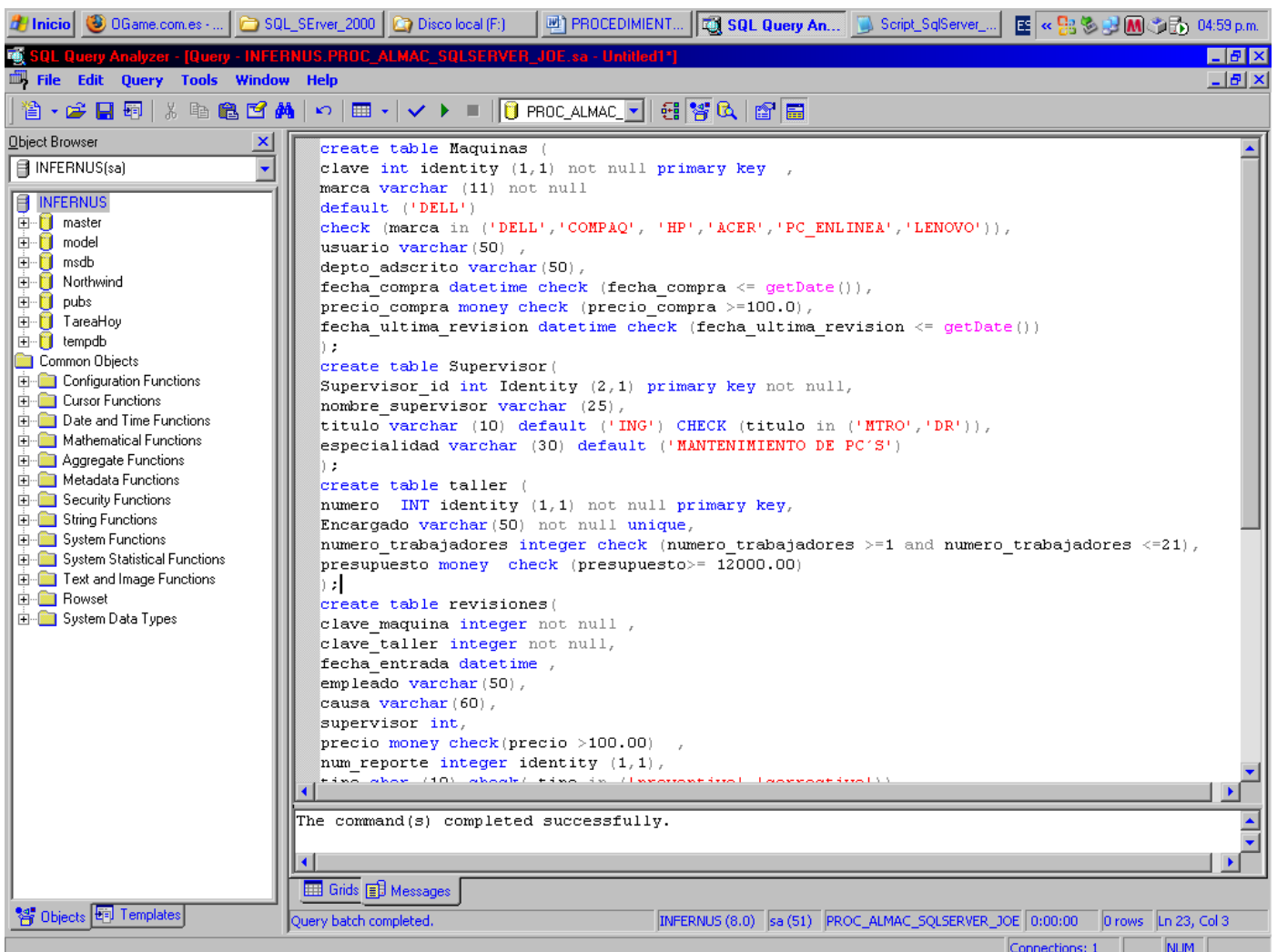


Creación de la base de Datos



Pulsamos F5 para Actualizar, y una vez posicionados en nuestra base de datos, se crean los scripts que serán soportados por el manejador para la creación de las tablas



DEL SCRIPT: **Script_SqlServer_Tarea3_sP_cREAbd**

EN LA SIG PAGINA SE MUESTRAN LOS SCRIPTS COMPLETOS

```

create table Maquinas (
clave int identity (1,1) not null primary key ,
marca varchar (11) not null
default ('DELL')
check (marca in ('DELL','COMPAQ', 'HP','ACER','PC_ENLINEA','LENOVO')),
usuario varchar(50) ,
depto_adscrito varchar(50),
fecha_compra datetime check (fecha_compra <= getDate()),
precio_compra money check (precio_compra >=100.0),
fecha_ultima_revision datetime check (fecha_ultima_revision <= getDate())
);

```

```

create table Supervisor(
Supervisor_id int Identity (2,1) primary key not null,
nombre_supervisor varchar (25),
titulo varchar (10) default ('ING') CHECK (titulo in ('MTRO','DR')),
especialidad varchar (30) default ('MANTENIMIENTO DE PC'S')
);

```

```

create table taller (
numero INT identity (1,1) not null primary key,
Encargado varchar(50) not null unique,
numero_trabajadores integer check (numero_trabajadores >=1 and numero_trabajadores <=21),
presupuesto money check (presupuesto>= 12000.00)
);

```

```

create table revisiones(
clave_maquina integer not null ,
clave_taller integer not null,
fecha_entrada datetime ,
empleado varchar(50),
causa varchar(60),
supervisor int,
precio money check(precio >100.00) ,
num_reporte integer identity (1,1),

```

```

tipo char (10) check( tipo in ('preventivo','correctivo')) ,
primary key (clave_maquina,clave_taller,fecha_entrada) ,

```

```

foreign key (clave_maquina) references Maquinas (clave)
on delete cascade
on update cascade,
foreign key (supervisor) references Supervisor (Supervisor_id),

```

```

foreign key (clave_taller) references taller (numero)
on delete cascade
on update cascade

```

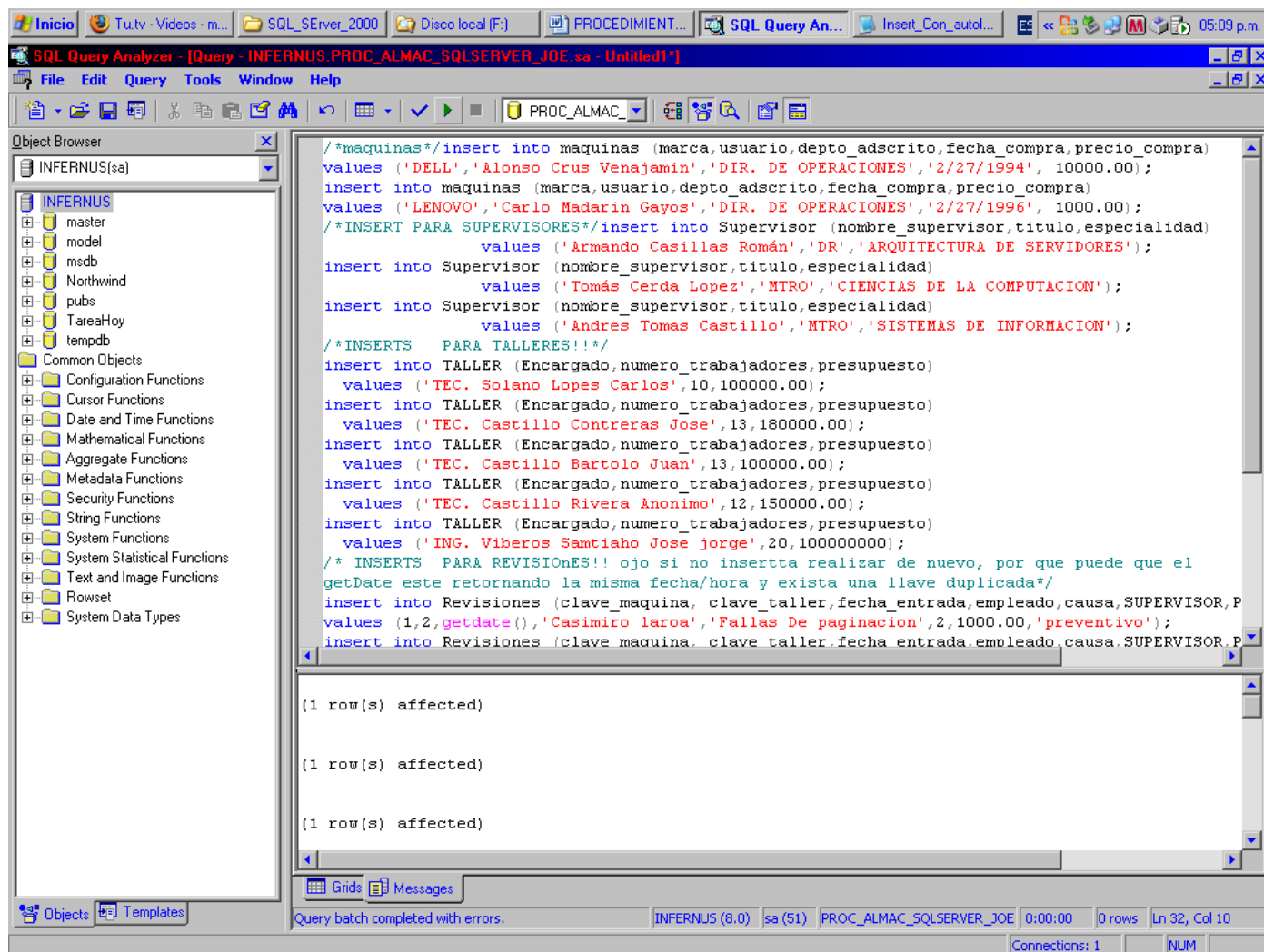
```
);
```

```

CREATE TABLE REPORTES (
maqu INT primary key,
total int,
foreign key (maqu) references MAQUINAS (clave)
on delete cascade
on update cascade
);

```

REALIZAR LAS INSERCCIONES EN LAS TABLAS GENERADAS,
(ver archivo **Insert_Con_autoIncrement_inserciones_para_lastablas**)



En la tabla MAQUINAS se insertan 6 columnas y el auto_increment automáticamente va generando la llave primaria para cada máquina

Se crea TALLERES que puedan atender mas de una maquina, a los cuales se les asigna un encargado y un presupuesto para poder atender la demanda.

Las revisiones están compuestas por la relacion entre las maquinas y los talleres, cada vez que una maquina va al taller se genera un registro de revision

La tabla REPORTES nos lleva un contro de cuantas veces una maquina ha ingresado al taller, independientemente del taller que sea.

Los supervisores funciona como auxiliares de las revisiones, es decir supervisan el trabajo que se hace en un taller por un determinado equipo.

A continuación se describe el procedimiento almacenao empleado en el ejercicio. Archiv: **SP_SinUpdate**

```

ERNU PROC_ALMAC_SQLSERVER_JOE.sa - Untitled1*]
low Help

create procedure MaquinariaConrevisión5
@Nuevamarca varchar(10),@NuevoUsuario varchar(50),@NuevoDepto varchar(50),--maquinaria
@Nencargado varchar(50),@NuevoCosto Money, @nuevaFecha datetime, --Reparacion
@NuevaCausa varchar(60),@Mamnto char(10)
as

declare @count as smallint
declare @Mid as SMALLINT
declare @Tid as smallint
declare @reg as int
declare @band as int

select @count = count (*)
from dbo.MAQUINAS
WHERE marca=@Nuevamarca and usuario=@NuevoUsuario and depto_adscrito=@NuevoDepto
if @count > 0
begin
    print 'Esta maquina ya está dada de alta'
    set @band = 1
    --return
end

```

1.-Se declaran los parámetros que serán mandados.

2.-Los parámetros manejados son columnas de la tabla maquinas, por lo que en el primer 'query', se determina si ya existe una maquina con esas columnas.

//se inicia transacción

3.-Si la maquina no con las características mandadas como datos al proc. Almacenado aun no existe en la base de datos, se realiza una inserción al registro maquinas.

4.- En el siguiente 'query', se obtiene a través de una consulta la llave de la maquina

5.- si la maquina no existe en la base de datos se hace un **roll back**

6.- se obtiene de igual manera el Taller (id), en base al nombre del encargado.

7.- se obtienen de la tabla de reportes de generado por la maquina analizada

8.- Esta bandera, permite decidir:

Si la maquina del registro es nuevo, se genera un nuevo registro en la tabla **Reportes**, que lleve el control de esta nueva maquina, si no simplemente realiza una actualización en **Reportes**

```

begin transaction /* inicia la transacción??*/

if @count <= 0
begin
    insert into dbo.MAQUINAS(marca,usuario,depto_adscrito,fecha_compra)
    values (@NuevaMarca,@NuevoUsuario,@NuevoDepto,getDate());
    PRINT 'SE AÑADIO UN REGISTRO A LA BASE DE DATOS'
end

select @Mid = clave
from dbo.MAQUINAS
where marca=@NuevaMarca and usuario=@NuevoUsuario and depto_adscrito=@NuevoDepto

if @Mid is null
begin
    print 'clave de Maquinaria no valida'
    rollback
    return
end

/*seleccionar la llave del taller encargado de la reparación*/

```

```

select @Tid = numero
from dbo.TALLER
where Encargado = @Nencargado
if @Tid is null
begin
    print 'clave de TALLER no valida'
    rollback
    return
end

select @reg = total
from dbo.Reportes
where maquina=@Mid
set @reg = @reg + 1

```

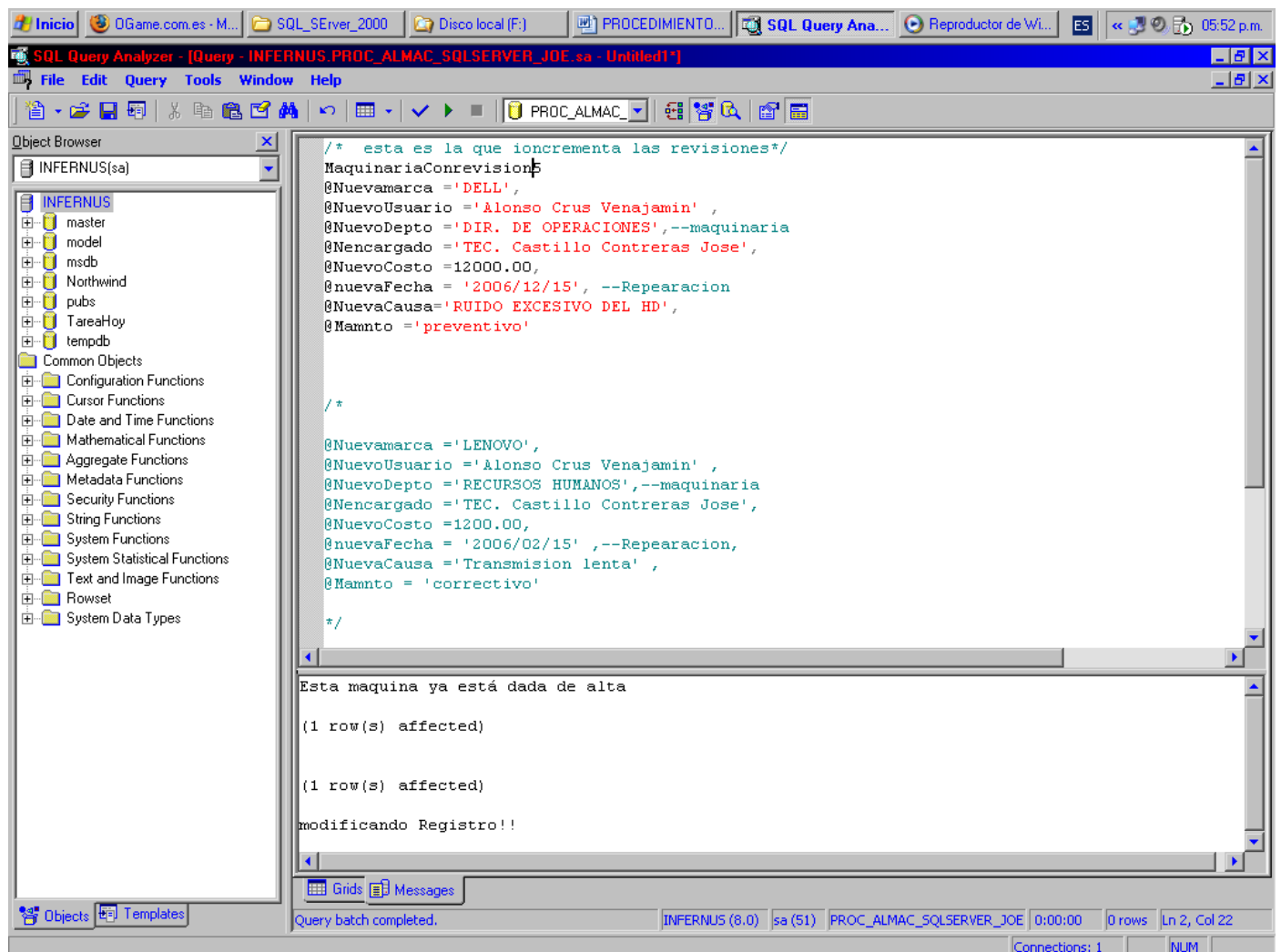
```

if @band = 1
begin
    insert into revisiones (clave_maquina,clave_taller,fecha_entrada,causa,supervisor,pre
    values (@Mid,@Tid,getDate(),@NuevaCausa,02,@NuevoCosto,@Mamnto)

    update dbo. REPORTES
    set TOTAL=@reg
    where @Mid = maquina
    print 'modificando Registro!!'
end
else
begin
    insert into revisiones (clave_maquina,clave_taller,fecha_entrada,causa,supervisor,pr
    values (@Mid,@Tid,getDate(),@NuevaCausa,02,@NuevoCosto,@Mamnto)
    | insert into reportes (maquina,total) values (@Mid,@reg)
end
commit
go

```

LLAMADA AL PROCEDIMIENTO ALMACENADO. (archivo: **tESTsCRIPt_sP_SINuPDATE**)



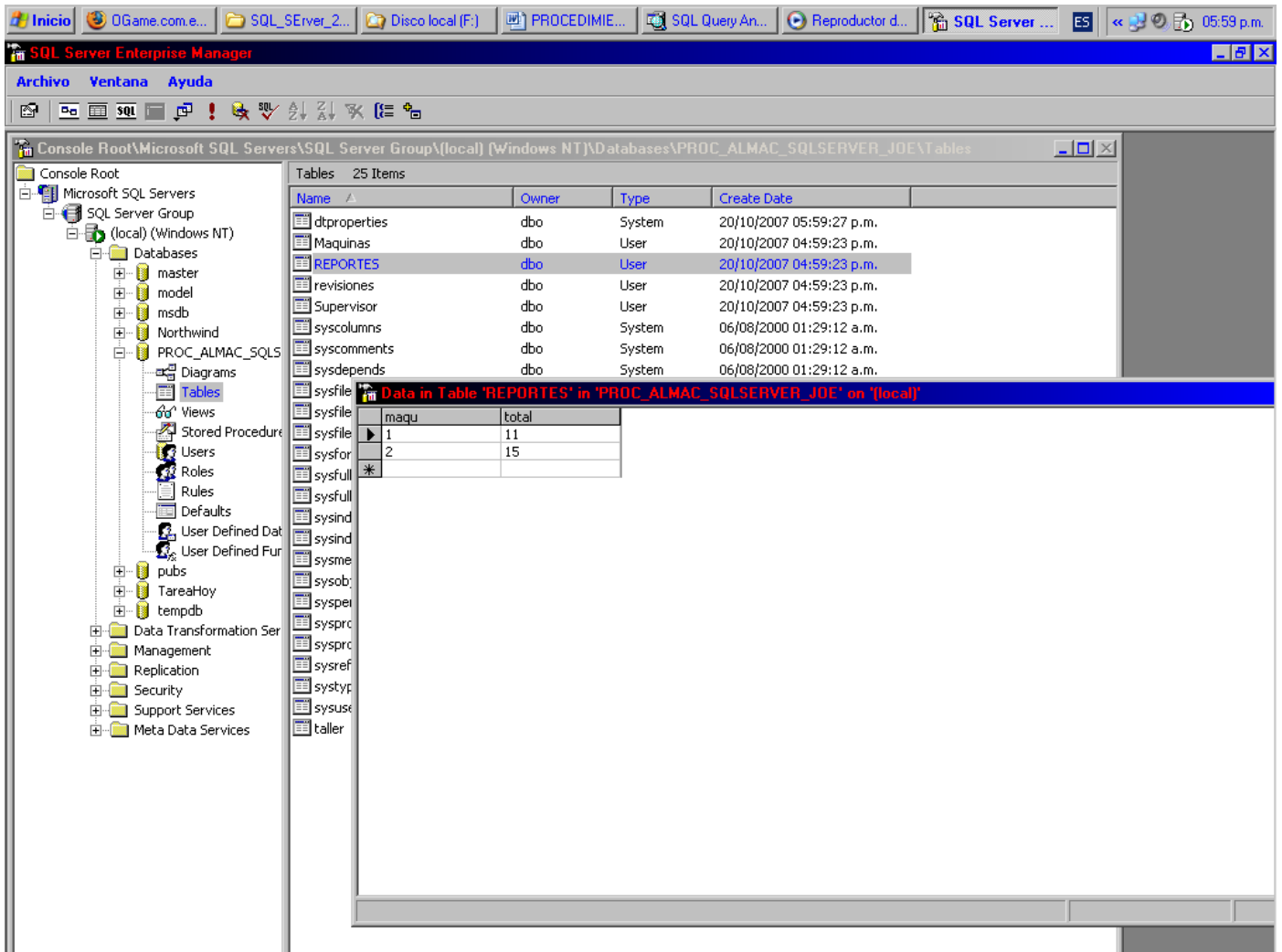
LOS DATOS QUE SE ENVIAN SON:

Marca, usuario, y el Depto, todos ellos pertenecen a las características de la tabla MAQUINAS, el nombre del Encargado servira para poder vincular la revision a un taller, costo, causa, matenimiento y fecha serviran para poder crear un resgitro en la tabla de REVISIONES.

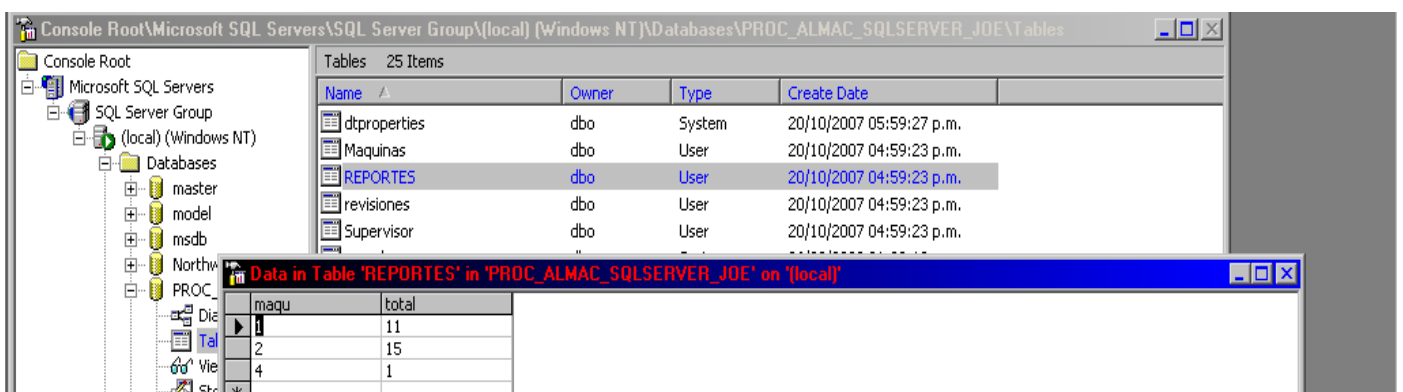
LAS CARACTERISTICAS INTRODUCIDAS COINCIDEN CON UNA MAQUINA EXISTENTE EN LA BASE DE DATOS, EL SP, OBTIENE EL ID DE ESA MAQUINA E INMEDIATAMENTE REALIZA UNA INSERCION DE ACTUALIZACION EN LA TABLA DE REPORTES.

VEAMOS COMO QUEDO LA TABLA DE REPORTE.

RECORDEMOS QUE EN AL INSERTAR EN LAS TABLAS, A LA MAQUINA UNO LE GENERAMOS UNA INSERCIÓN DE 10 REPORTE, (ver script de inserciones), AHORA ESE REGISTRO FUE MODIFICADO



Ahora analizaremos un caso donde las características no coincidan con ninguna de las maquinas hasta ahora introducidas en la base de datos, entonces el porcedimiento almacenado generara un nuevo registro en la tabla de maquinas, seguido creara un registro en la tabla de revisiones, y finalmente creara un historial de esa nueva maquina en la tabla de reportes.



CREACION DE TRIGGERS

Para efectos de clase el análisis del uso de triggers lo haremos sobre la misma base de datos en la que trabajamos los procedimientos almacenados, a fin de que puedan hacerse una analogía que permita diferenciar entre uno y otro.

Nota (no es necesario crear SP antes, los triggers son independientes y pueden funcionar con solo crear la Base de Datos y realizar las inserciones en ella)

Crearemos un trigger a partir del sig. Script: **scrip_tigger.txt**

A continuación analizaremos el código del Script:

```
create trigger Nuevo_reporte2 on
dbo.Revisiones
for insert
as
declare @count as smallInt
declare @exis as smallInt
declare @cuenta as smallInt

select @count = clave_maquina from inserted
-- from dbo.revisiones
print 'cadena -..' + convert(char(1),@count )

select @exis = maqu,@cuenta=total
from dbo.reportes
      where @count=maqu

if @exis >0
begin
  update dbo. REPORTES
  set TOTAL=@cuenta+1
  where @exis =maqu
  print 'modificando Registro!!'
end
else
begin
  insert into dbo.REPORTES values (@count,1 )
end
```

Las palabras create trigger indican al manejador que se esta creando un trigger sobre la base de datos.

La palabra **on** indica en especifico sobre que tabla se esta creando el trigger.

Declaramos unas variables que servirán del sig. Modo.

@count = clave de la maquina que se insertó

@exis = clave de esa maquina en la tabla de REPORTES

@cuenta= numero de reportes de esa maquina en la tabla de REPORTES

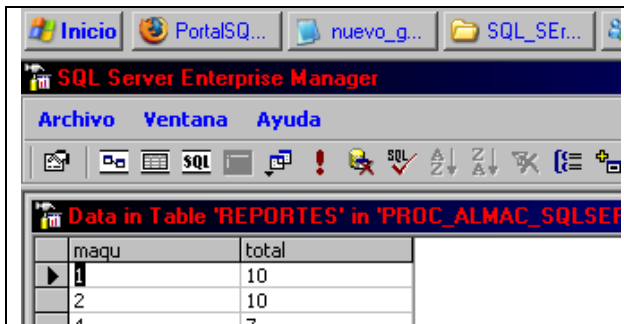
La idea general de este trigger, es realizar un disparo cada vez que una maquina llegue a revisiones, a fin de llevar un control de cuantas veces esta maquina ha estado en algún taller.

Los controles de flujo, sirven para determinar la accion a realizar según la clave de la maquina, es decir si la maquina insertada ya tiene un historial habra que actualizarlo, y de no ser así, se crear un nuevo historial para esta maquina.

Para probar el funcionamiento de este trigger, mandaremos una inserción a la tabla de REVISIONES, primero con un valor que ya existe para que el trigger actualice en la tabla de REPORTES, y después una máquina nueva para que el trigger le cree un nuevo historial a esa maquina en la tabla de REPORTES.

CONSULTA:

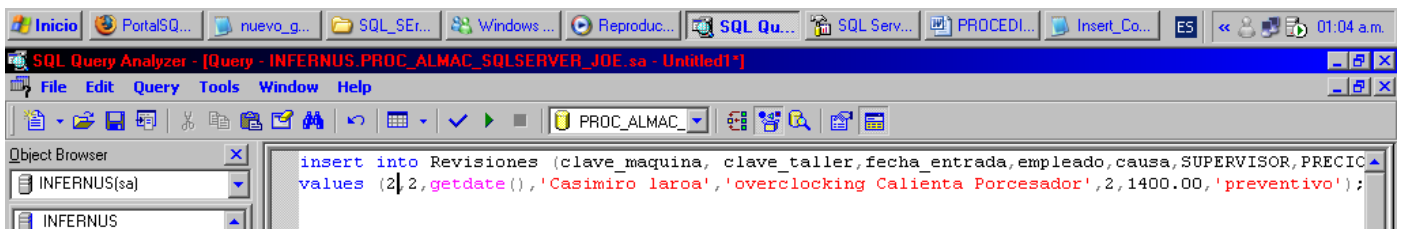
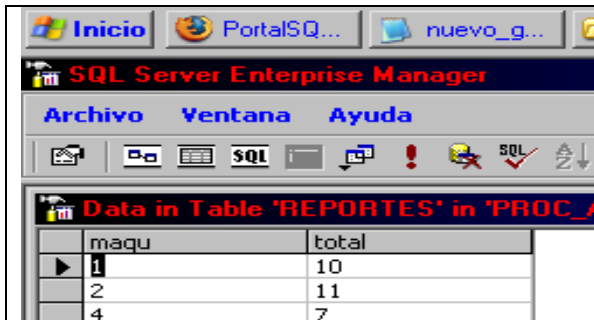
```
insert into Revisiones (clave_maquina,
clave_taller,fecha_entrada,empleado,causa,SUPERVISOR,PRECIO,tipo)
values (1,2,getdate(),'Casimiro laroa','Fallas De paginacion',2,1000.00,'preventivo');
```



maqu	total
1	10
2	10
4	7

Tenemos 10 reportes para la maquina dos antes de realizar la inserción.

Ejecutamos la inserción, y el trigger automáticamente se dispara

maqu	total
1	10
2	11
4	7

El disparador actualiza en la tabla de REPORTES, y ahora el total de reportes de la maquina dos son 11.