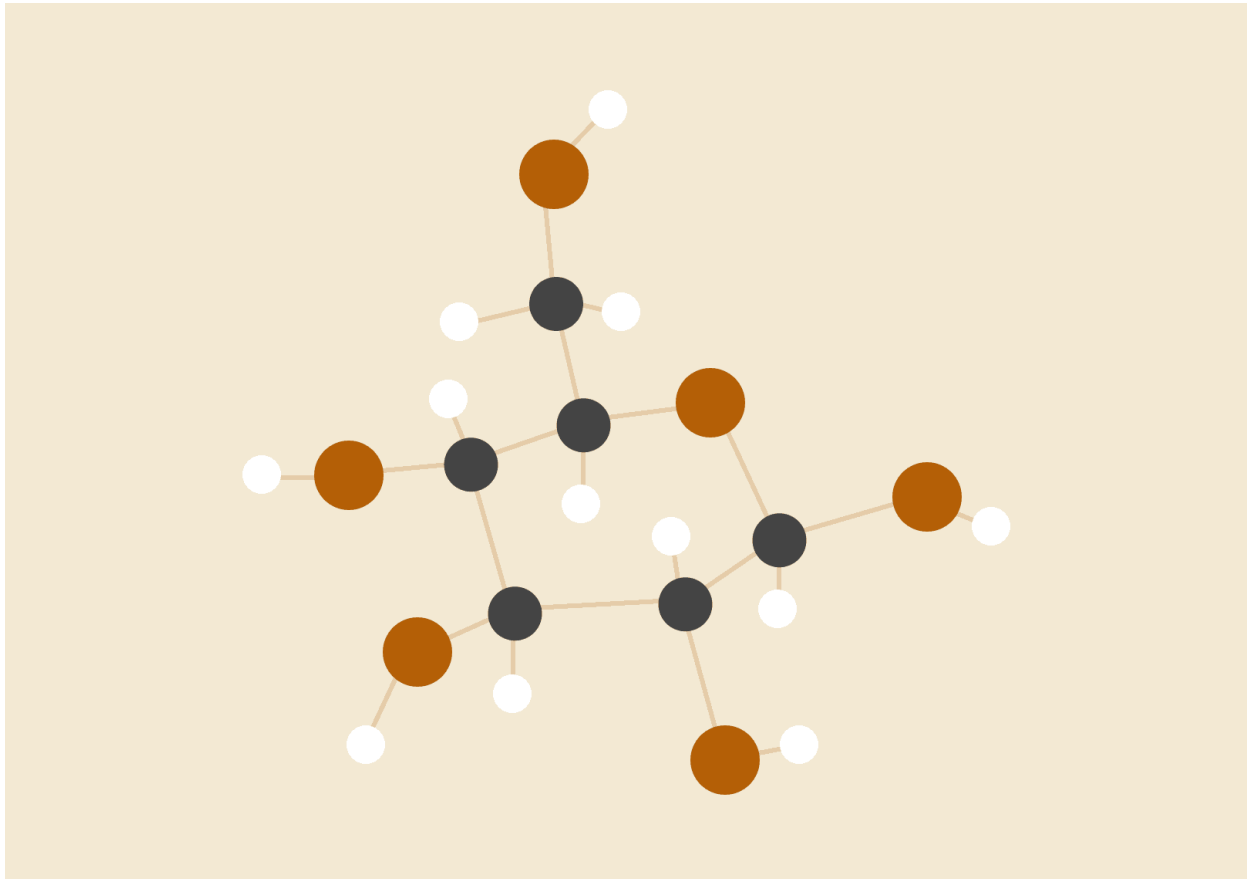


# Departamento de Tecnología Virtual:

## proyecto de diseño de Bases de Datos a partir de un ejemplo práctico



### Trabajo Académico BDA

Realizado por Adrián Rico, Juan Tomás y Marc Vicedo

Curso 2022/2023 - Ciencia de Datos

# Índice

→ Introducción. Enunciado del problema.....	3
→ Diagrama de clases.....	5
→ Requisitos de proceso y consulta.....	6
→ Esquema relacional.....	7
→ Restricciones de integridad.....	10
→ Diagrama Oracle.....	11
→ Carga de datos.....	12
→ Elaboración de informes.....	18
→ Programación de módulos.....	22



# Introducción. Enunciado del problema

El objetivo de este trabajo es diseñar e implantar una base de datos a partir de unos requisitos concretos, y que, al mismo tiempo, nos permita introducir datos de manera correcta y trabajar con ellos. Para ello utilizaremos el lenguaje SQL desde un servidor Oracle, en el que guardaremos nuestra base de datos.

Tomamos como ejemplo esta tarea en la que se nos pide realizar un diagrama de clases a partir de la solicitud de un departamento ficticio: el Departamento de Tecnología Virtual. La solicitud era la siguiente:

*El departamento de tecnología virtual y computación fantástica precisa de una base de datos para informatizar el proceso de gestión de sus investigadores.*

*El departamento está organizado en grupos de investigación que se identifican por un nombre que es único y de los que además se almacenará la dirección web del grupo, una descripción, el día y hora de la semana en la que se reúnen habitualmente y los investigadores que lo componen.*

*Los proyectos de investigación que se llevan a cabo en el departamento se identifican por un código, y deben de tener una descripción, una fecha de inicio, duración, y una cuantía económica. Estos siempre tienen un investigador principal que solamente puede ser un profesor. Además, podrán tener otros investigadores que sean profesores o becarios del ministerio como participantes en el proyecto. Se dice que un proyecto es del grupo de investigación al que pertenece el investigador principal, indistintamente de la procedencia del resto de investigadores.*

*De los investigadores del departamento se conoce el DNI que los identifica, nombre, dirección, email, grupo al que pertenecen, y un conjunto de números de teléfono. Entre los investigadores diferenciaremos a los profesores, a los becarios del ministerio y a los becarios de proyecto. De los becarios del ministerio será necesario conocer su tutor que será un profesor del mismo grupo, y la fecha de incorporación. Los becarios de proyecto son becarios que participan siempre y de manera exclusiva en un proyecto de investigación debiendo almacenar un informe de su labor a desarrollar en el proyecto.*

*Cuando se produce un gasto en un proyecto éste se anota a la contabilidad del departamento. Los gastos se identifican por un número que es único dentro del departamento y debe quedar reflejado además del proyecto, la cuantía del gasto, la descripción y el investigador que lo ha realizado. Solamente los profesores y becarios del ministerio que participan en el proyecto pueden realizar gastos.*

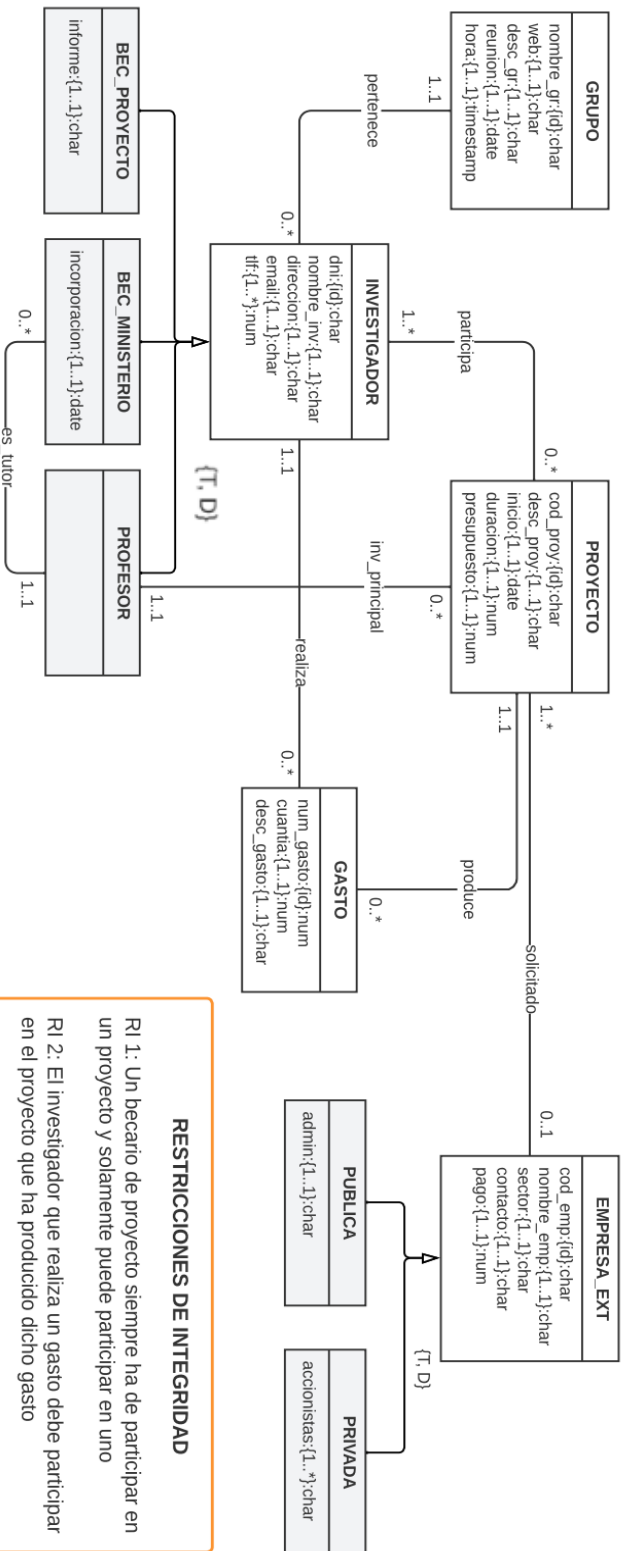
En el enunciado del problema, hemos marcado todas palabras clave que se traducirán en **futuros atributos** de nuestra base de datos. Como ejemplo, podemos centrarnos en el último párrafo: un gasto tiene diversos atributos relacionados a él, tales como su número identificador, el proyecto en el que se ha generado o su cuantía.

Además, a modo de ampliación, hemos añadido un nuevo objeto de información al diagrama: las **empresas externas** (EMPRESA\_EXT). Entendemos que hay empresas que pueden solicitar al Departamento proyectos relevantes para su sector. Dentro de este objeto, especificamos un **código** que identifique a la empresa, su **nombre**, el **sector** al que pertenece, un **contacto** de dicha empresa y el **pago que se efectúa** por llevar a cabo el proyecto. Una empresa puede solicitar o no un proyecto; sin embargo, en nuestra base de datos solamente incluiremos las empresas que hayan solicitado proyectos. Asimismo, siguen habiendo proyectos independientes, que no han sido solicitados específicamente por empresas. Dentro del objeto que hemos diseñado, creamos una especialización que indique **si esta empresa es pública o privada**. En el primer caso, se incluirá la **administración pública** de la que depende esta empresa; en el segundo caso, aparecerá el conjunto de **accionistas** que la respaldan.

En la siguiente página adjuntamos el **diagrama de clases** correspondiente que hemos diseñado para este problema, incluyendo las restricciones de integridad que no pueden incluirse dentro de dicho diagrama.

# Diagrama de clases

## Departamento de Tecnología Virtual Trabajo académico: Tarea 1 | Adrián Rico, Juan Tomás y Marc Viceo



### RESTRICCIONES DE INTEGRIDAD

- RI 1: Un becario de proyecto siempre ha de participar en un proyecto y solamente puede participar en uno
- RI 2: El investigador que realiza un gasto debe participar en el proyecto que ha producido dicho gasto
- RI 3: Los gastos solo pueden ser realizados por investigadores que sean profesores o becarios del ministerio
- RI 4: El tutor tiene que pertenecer al mismo grupo que su becario del ministerio tutelado

# Requisitos de proceso y consulta

Ahora, pasamos a redactar una **lista de requisitos** que podrían ser aplicados a la base de datos una vez esté completada:

1. Obtener el código de todos los proyectos asignados a un grupo
2. Obtener el DNI y nombre de todos los investigadores de un grupo
3. Obtener el código de los proyectos en los que participe más de un becario de proyecto
4. Obtener la cuantía total del gasto producido por un proyecto, además de su presupuesto
5. Obtener el código de los proyectos que hayan sido solicitados por alguna empresa pública
6. Obtener el DNI y nombre de los investigadores que participen en proyectos que no estén asignados a su grupo
7. Obtener el DNI, nombre y conjunto de teléfonos de los investigadores principales de proyectos solicitados por empresas privadas
8. Obtener el nombre de los grupos en los que todas las personas que formen parte de ellos sean profesores
9. Borrar todos los proyectos que tengan un presupuesto de más de 10000€
10. Borrar todos los grupos que no tengan proyectos asignados
11. Añadir un nuevo proyecto, “Cohete”, donde se intentará buscar la tecnología para desarrollar un cohete español. Se iniciará el 1 de enero de 2023, y durará 6 meses (es decir, 180 días), con un presupuesto inicial de 500€. Su investigador principal será el profesor Marc Vicedo, y por tanto este proyecto se le asignará a su grupo
12. Añadir una nueva empresa pública, la Agencia Espacial Española (AEE), dependiente del Ministerio de Ciencia e Innovación, ya que ha solicitado el proyecto “Cohete”, por el que pagará 2000€. Es una empresa del sector aeroespacial, y su contacto es Pedro Duque.

# Esquema relacional

A continuación, desarrollamos el esquema relacional de nuestra base de datos, explicando las claves correspondientes a cada relación. Cabe destacar que hemos comprobado previamente que este esquema está en **tercera forma normal** (3FN), ya que sus atributos únicamente pueden tomar valores atómicos (es decir, valores simples e indivisibles), todos los atributos no primos dependen de las claves correspondientes de sus respectivas relaciones, y no existen dependencias funcionales entre atributos no primos.

---

**GRUPO**(nombre\_gr:char, web:char, desc\_gr:char, reunion:varchar, hora:varchar)  
CP:{nombre\_gr}  
VNN: {web, desc\_gr, reunion, hora}

**PROYECTO**(cod\_proy:char, desc\_proy:char, inicio:date, duracion:num, presupuesto:num, empresa:char, inv\_principal:char)  
CP: {cod\_proy}  
VNN: {desc\_proy, inicio, duracion, presupuesto, inv\_principal}  
CAj: {empresa} -> EMPRESA\_EXT(cod\_emp)  
Borrado en cascada, modificación en cascada  
CAj: {inv\_principal} -> PROFESOR(dni)  
Borrado en cascada, modificación en cascada

**INVESTIGADOR**(dni:char, nombre\_inv:char, direccion:char, email:char, grupo:char)  
CP: {dni}  
VNN: {nombre\_inv, direccion, email, grupo}  
CAj: {grupo} -> GRUPO(nombre\_gr)  
Borrado restrictivo, modificación en cascada

**TELEFONO\_INV**(dni:char, telefono:num)  
CP: {telefono}  
VNN: {dni}  
CAj: {dni} -> INVESTIGADOR(dni)  
Borrado en cascada, modificación en cascada

**PARTICIPA**(dni:char, proyecto:char)

CP: {dni, proyecto}

CAj: {dni} -> INVESTIGADOR(dni)

Borrado en cascada, modificación en cascada

CAj: {proyecto} -> PROYECTO(cod\_proy)

Borrado en cascada, modificación en cascada

**BEC\_PROYECTO**(dni:char, informe:char)

CP: {dni}

VNN: {informe}

CAj: {dni} -> INVESTIGADOR(dni)

Borrado en cascada, modificación en cascada

**BEC\_MINISTERIO**(dni:char, incorporacion:date, tutor:char)

CP: {dni}

VNN: {incorporacion, tutor}

CAj: {dni} -> INVESTIGADOR(dni)

Borrado en cascada, modificación en cascada

CAj: {tutor} -> PROFESOR(dni)

Borrado restrictivo, modificación en cascada

**PROFESOR**(dni:char)

CP: {dni}

CAj: {dni} -> INVESTIGADOR(dni)

Modificación y borrado en cascada

**GASTO**(proyecto:char, dni:char, num\_gasto:num, cuantia:num, desc\_gasto:char)

CP: {num\_gasto}

VNN: {cuantia, desc\_gasto, proyecto, dni}

CAj: {proyecto} -> PROYECTO(cod\_proy)

Borrado restrictivo, modificación en cascada

CAj: {dni} -> INVESTIGADOR(dni)

Borrado restrictivo, modificación en cascada



*Los siguientes atributos corresponden a la ampliación del enunciado del problema, explicada en la introducción del informe*

**EMPRESA\_EXT**(cod\_emp:char, nombre\_emp:char, sector:char, contacto:char, pago:num)

CP: {cod\_emp}

VNN: {nombre\_emp, sector, contacto, pago}

**EM\_PUBLICA**(empresa:char, admin:char)

CP: {empresa}

VNN: {admin}

CAj: {empresa} -> EMPRESA\_EXT(cod\_emp)

Borrado en cascada, modificación en cascada

**EM\_PRIVADA**(empresa:char)

CP: {empresa}

CAj: {empresa} -> EMPRESA\_EXT(cod\_emp)

Borrado en cascada, modificación en cascada

**ACCIONISTA\_PRIV**(empresa:char, nombre:char)

CP: {empresa, nombre}

CAj: {empresa} -> EM\_PRIVADA(empresa)

Borrado en cascada, modificación en cascada

---

En total trabajaremos con 13 relaciones en nuestra base de datos, ya que hemos necesitado construir **tres tablas nuevas**: una para la relación PARTICIPA (en la que aparecen las claves ajenas de INVESTIGADOR y PROYECTO correspondientes), y dos para los “conjuntos de datos”; por un lado los teléfonos de los investigadores (TELEFONO\_INV), y por otro los nombres de los accionistas de empresas privadas (ACCIONISTA\_PRIV). Además, hemos introducido un cambio en la relación GRUPO, donde los atributos *reunion* y *hora* se han cambiado a formato Varchar, porque la carga de datos no sería posible sin esta modificación.

# Restricciones de integridad

Listamos ahora las **restricciones de integridad** de nuestro esquema:

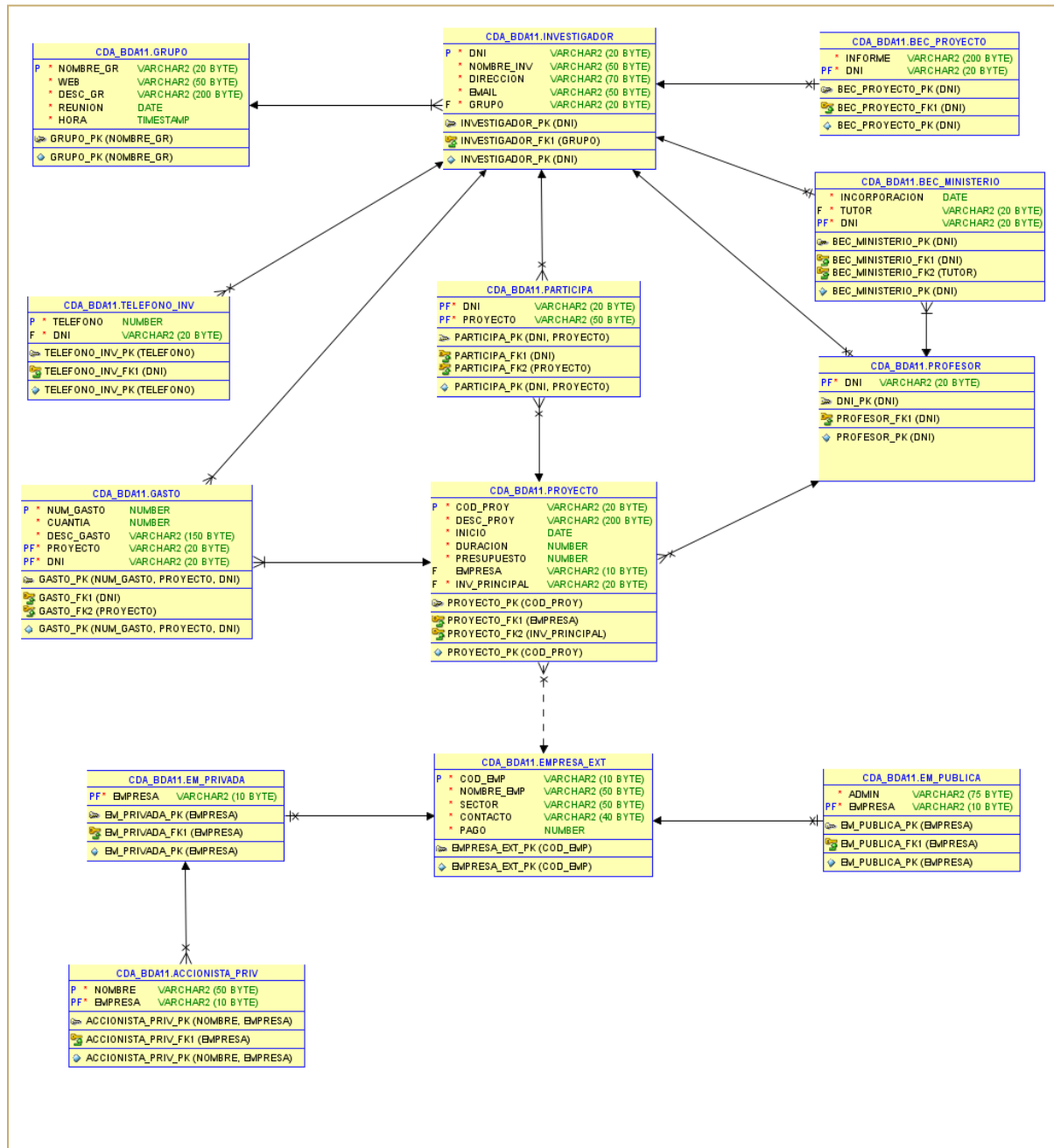
- RI 1: Todo valor de *dni* de la relación BEC\_PROYECTO aparece una única vez en el atributo *dni* de PARTICIPA
- RI 2: Todo valor de *dni* de la relación GASTO aparece en el atributo *dni* de PARTICIPA, y el atributo *proyecto* de esta relación debe coincidir con el atributo *proyecto* de la anterior para los valores de *dni* dados
- RI 3: Ningún valor de *dni* de la relación GASTO puede aparecer en el atributo *dni* de BEC\_PROYECTO
- RI 4: Para todo valor de *dni* y *tutor* de la relación BEC\_MINISTERIO aparece en el atributo *dni* de INVESTIGADOR, y el atributo *grupo* debe coincidir para ambos valores de *dni* en esta relación

Más allá de las restricciones arriba establecidas (las que ya aparecían en el diagrama de clases), vemos necesario añadir algunas más:

- RI 5: Todo valor de *cod\_emp* de la relación EMPRESA\_EXT aparece al menos una vez en el atributo *empresa* de PROYECTO
- RI 6: No existe ningún valor de *dni* de la relación INVESTIGADOR que no aparezca en las relaciones BEC\_MINISTERIO, BEC\_PROYECTO o PROFESOR.
- RI 7: No aparece ningún valor del atributo *dni* igual en las relaciones BEC\_MINISTERIO, BEC\_PROYECTO y PROFESOR.
- RI 8: No existe ningún valor de *cod\_emp* de la relación EMPRESA\_EXT que no aparezca en el atributo *empresa* de las relaciones PUBLICA o PRIVADA.
- RI 9: No aparece ningún valor de *empresa* igual en las relaciones PUBLICA y PRIVADA.
- RI 10: Todo valor de *dni* de la relación INVESTIGADOR debe aparecer en el atributo *dni* de la relación TELEFONO\_INV.
- RI 11: Todo valor de *empresa* de la relación PRIVADA debe aparecer en el atributo *empresa* de la relación ACCIONISTA\_PRIV.

# Diagrama Oracle

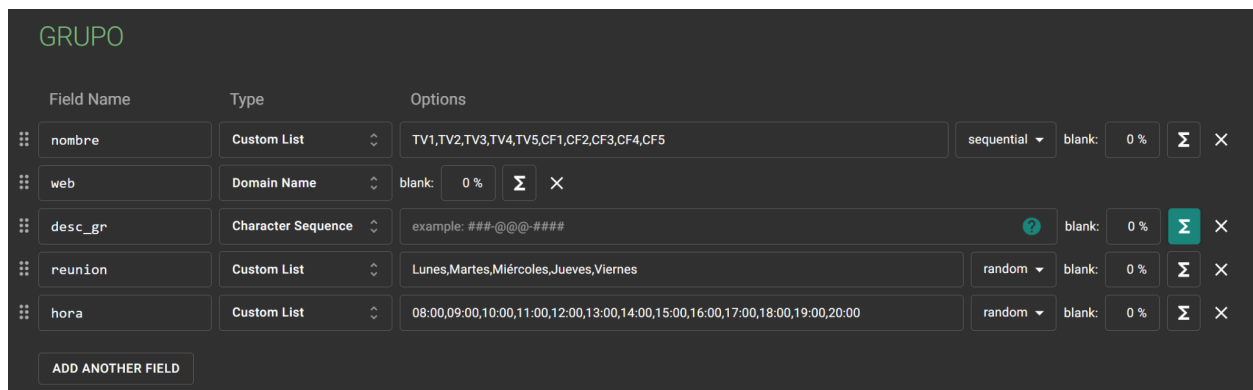
Adjuntamos el diagrama Oracle de nuestra base de datos, generado a partir de la herramienta **Data Modeler**:



# Carga de datos

Después de haber diseñado la estructura que tendrá nuestra base de datos relacional, es el momento de **introducir datos** dentro de ella para así poder realizar diversos informes y procedimientos. Algunos datos se han introducido de manera manual, aunque para la mayoría hemos utilizado herramientas *online* de generación masiva de datos como **Mockaroo**.

En primer lugar, hemos creado los datos de la **relación GRUPO**:



The screenshot shows the Mockaroo interface for generating data for a table named "GRUPO". The interface has a dark theme. At the top, the table name "GRUPO" is displayed in green. Below it, there is a table with columns: Field Name, Type, and Options. The table contains five rows of field definitions:

Field Name	Type	Options
nombre	Custom List	TV1,TV2,TV3,TV4,TV5,CF1,CF2,CF3,CF4,CF5
web	Domain Name	blank: 0 %
desc_gr	Character Sequence	example: ###-@@-####
reunion	Custom List	Lunes,Martes,Miércoles,Jueves,Viernes
hora	Custom List	08:00,09:00,10:00,11:00,12:00,13:00,14:00,15:00,16:00,17:00,18:00,19:00,20:00

Each row has a dropdown menu for the "Options" column, showing the generation method: "sequential" for "nombre", "blank" for "web", "example" for "desc\_gr", "random" for "reunion", and "random" for "hora". To the right of each row, there are controls for "blank" (0 %), a "Generate" button (Σ), and a "Delete" button (X). At the bottom left, there is a button labeled "ADD ANOTHER FIELD".

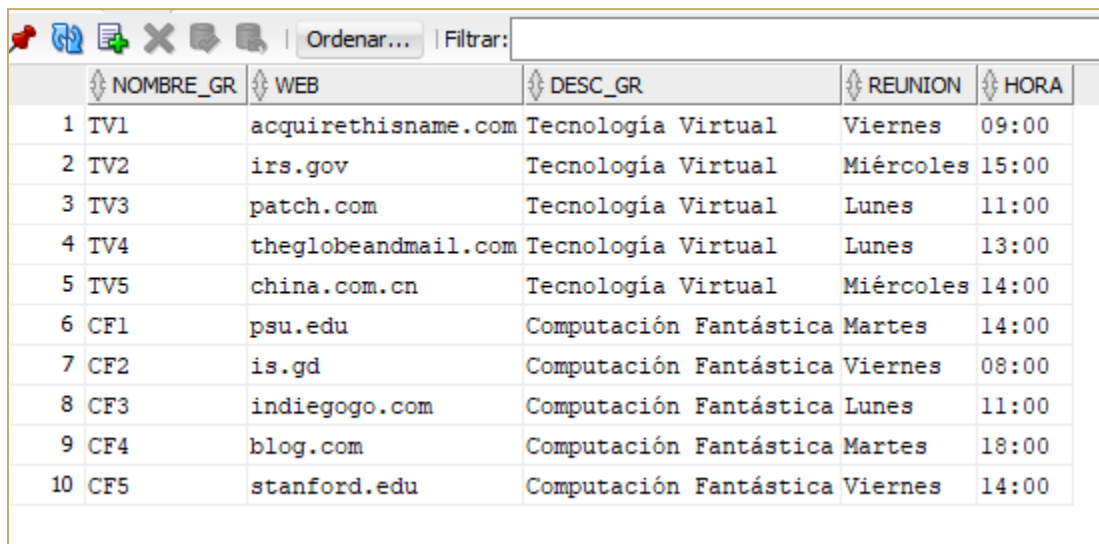
En nuestra base de datos tendremos 10 grupos, 5 de cada sector del departamento (los que empiezan por TV son de Tecnología Virtual, los que empiezan por CF son de Computación Fantástica). En este caso, tal y como se ve en la imagen superior, la generación de datos será secuencial y no aleatoria, para asegurar que se formarán 10 filas, una con cada nombre de nuestra lista.

En el atributo *web*, se generarán dominios web aleatorios, y en los dos últimos apartados (*reunion* y *hora*), hemos cambiado el tipo de dato a Varchar para facilitar la legibilidad, ya que generar una fecha aleatoria en formato SQL no permitía reflejar que se celebrase una reunión semanal. Por ello, creamos una lista de valores con los días de la semana que, en este caso sí, se generarán de forma aleatoria.

Por último, el atributo *desc\_gr* incluye una fórmula para asegurar que se escribe ‘Tecnología Virtual’ o ‘Computación Fantástica’ según su sector, tal y como podemos ver en la imagen inferior.

```
if nombre == 'TV1' or
nombre == 'TV2' or
nombre == 'TV3' or
nombre == 'TV4' or
nombre == 'TV5' then 'Tecnología Virtual'
else 'Computación Fantástica' end
```

La relación finalmente quedaría de la siguiente manera:



	NOMBRE_GR	WEB	DESC_GR	REUNION	HORA
1	TV1	acquirethisname.com	Tecnología Virtual	Viernes	09:00
2	TV2	irs.gov	Tecnología Virtual	Miércoles	15:00
3	TV3	patch.com	Tecnología Virtual	Lunes	11:00
4	TV4	theglobeandmail.com	Tecnología Virtual	Lunes	13:00
5	TV5	china.com.cn	Tecnología Virtual	Miércoles	14:00
6	CF1	psu.edu	Computación Fantástica	Martes	14:00
7	CF2	is.gd	Computación Fantástica	Viernes	08:00
8	CF3	indiegogo.com	Computación Fantástica	Lunes	11:00
9	CF4	blog.com	Computación Fantástica	Martes	18:00
10	CF5	stanford.edu	Computación Fantástica	Viernes	14:00

---

A continuación hemos creado los datos de la **relación INVESTIGADOR**, donde todos los atributos han sido generados aleatoriamente: por ejemplo, el valor de *dni* es una cadena de caracteres con ocho cifras aleatorias y una letra mayúscula aleatoria, siguiendo los comandos de expresiones regulares utilizados por Mockaroo. En la siguiente página podemos ver el trabajo de carga de datos y el resultado final dentro de SQL.

**INVESTIGADOR**

Field Name	Type	Options
dni	Character Sequence	#####* ? blank: 0 % Σ ×
nombre_inv	Full Name	blank: 0 % Σ ×
direccion	Street Address	blank: 0 % Σ ×
email	Email Address	blank: 0 % Σ ×
grupo	Custom List	TV1,TV2,TV3,TV4,TV5,CF1,CF2,CF3,CF4,CF5 random blank: 0 % Σ ×

ADD ANOTHER FIELD

	DNI	NOMBRE_INV	DIRECCION	EMAIL	GRUPO
1	58725374W	Emerson Waren	660 Fuller Pass	ewaren2p@archive.org	CF3
2	14572352V	Danna Symms	87197 Dennis Pass	dsymms2q@google.co.uk	TV4
3	01636447K	Clarita Denyer	3 Fair Oaks Point	cdenyer2r@so-net.ne.jp	CF1
4	22104768Y	Julietta Quadling	8654 Glacier Hill Drive	jquadling0@weibo.com	TV3
5	132869490	Rosco MacArthur	455 Green Hill	rmacarthur1@wired.com	CF5
6	580952590	Vikki Jakolevitch	6323 Warner Lane	vjakolevitch2@t-online.de	CF3
7	99721803B	Martie Ilden	220 Swallow Circle	milden3@pinterest.com	CF4
8	73504907K	Even Sawl	2624 Menomonie Plaza	esawl4@cornell.edu	TV4
9	56220429L	Gaultiero Blaymires	099 Farmco Avenue	gblaymires5@dedecms.com	CF4
10	37276989K	Drona Brito	5 Bay Crossing	dbrito6@wisc.edu	TV3
11	25766618Y	Rheba Stovles	8 Westerfield Hill	rstovles7@gnu.org	TV5

Para facilitar el diseño de posteriores relaciones, hemos incluido 100 investigadores (100 filas), de los cuales 70 serán profesores, 15 serán becarios de proyecto y los 15 restantes serán becarios de ministerio. Los datos de los atributos *dni* de dichas subclases han sido introducidos de forma manual; los 70 primeros se han introducido en la **relación PROFESOR**, y los 30 siguientes irán a cada uno de los becarios. Hemos realizado esta distribución para facilitar el cumplimiento de todas las claves ajenas, así como de las correspondientes restricciones de unicidad que no pueden ser violadas.

Para las relaciones **BEC\_MINISTERIO** y **BEC\_PROYECTO** también hemos utilizado generación aleatoria con Mockaroo:

BEC\_MINISTERIO

Field Name	Type	Options		blank:	0 %	Σ	×
dni	Character Sequence	0	?	blank:	0 %	Σ	×
tutor	Character Sequence	1	?	blank:	0 %	Σ	×
incorporacion	Datetime	01/01/2023 to 12/31/2023 format: SQL datetime		blank:	0 %	Σ	×

ADD ANOTHER FIELD

BEC\_PROYECTO

Field Name	Type	Options		blank:	0 %	Σ	×
dni	Character Sequence	0	?	blank:	0 %	Σ	×
informe	Custom List	Buen trabajo,Trabajo excelente,Necesita mejorar,No encaja en el proyecto	random	blank:	0 %	Σ	×

ADD ANOTHER FIELD

Tal y como hemos explicado anteriormente, los valores referidos a los DNI de los becarios se introducirán desde el SQL, 15 en cada relación. En cambio, el atributo *incorporacion* consistirá en una fecha aleatoria del año 2023, y el atributo *informe* será un valor aleatorio de entre las opciones que aparecen en la segunda imagen (Necesita mejorar, Buen trabajo, etc.). Este es el resultado final de ambas relaciones:

	INCORPORACION	TUTOR	DNI
1	06/12/2023	08181286N	17899246K
2	20/12/2023	47496894W	62644454L
3	17/09/2023	25766618Y	58991709K
4	21/06/2023	132869490	96685264E
5	05/02/2023	37276989K	83538312K
6	21/02/2023	96141534I	28930832C
7	11/07/2023	09419667R	42484516S
8	29/10/2023	99992765Q	88965095T
9	26/02/2023	78561864R	31783971H
10	09/12/2023	30587748I	70374100W
11	23/10/2023	37122857K	35444246V
12	21/08/2023	57343549M	81812743R
13	14/05/2023	59495694T	31172947G
14	15/08/2023	43138033N	05619349R
15	13/09/2023	81112022D	92077459T

	INFORME	DNI
1	Trabajo excelente	74427528J
2	Necesita mejorar	84509890H
3	Trabajo excelente	99033805U
4	Necesita mejorar	57039715C
5	Buen trabajo	21833542X
6	Trabajo excelente	96822803X
7	Buen trabajo	22373943S
8	Necesita mejorar	63435235E
9	Necesita mejorar	53949357H
10	Necesita mejorar	32772899H
11	Buen trabajo	91194873P
12	Necesita mejorar	63263898F
13	Trabajo excelente	58725374W
14	No encaja en el proyecto	14572352V
15	Necesita mejorar	01636447K






De la misma forma, para la **relación TELEFONO\_INV** utilizamos una estrategia mixta: por un lado generamos los números de teléfono aleatoriamente (números de nueve cifras, siendo la primera 6 siempre), y por otro los DNI correspondientes serán introducidos de forma manual. Para reflejar la distribución {1..\*} del atributo *teléfono*, generamos 150 filas e introducimos todos los valores de *dni* de la relación INVESTIGADOR, repitiendo 50 de ellos una vez más.

TELEFONO\_INV

Field Name	Type	Options
dni	Character Sequence	0 blank: 0 %
telefono	Character Sequence	6##### blank: 0 %

ADD ANOTHER FIELD

Para crear la **relación PROYECTO** hemos ideado cinco potenciales proyectos, que hemos introducido a mano en SQL. En la siguiente imagen podemos ver la descripción de cada uno de los proyectos, así como el resto de datos:

<div><div></div><div>Ordenar...</div><div>Filtrar:</div><div></div><div>▼ Accion</div></div>						
COD_PROY	DESC_PROY	INICIO	DURACION	PRESUPUESTO	EMPRESA	INV_PRINCIPAL
1 BDA	Crear una base de datos	01/01/2024	30	1000 (null)		25417706S
2 Fuentes	Crear un proyecto de fuente inteligente	01/01/2024	90	2000 AVL		99992765Q
3 PESTEL	Programar una aplicación para hacer el análisis PESTEL de una empresa	01/01/2024	180	20000 FCOMP		47496894W
4 Botellas	Averiguar qué modelo de producción de botellas es más eficiente	01/01/2024	30	1000 (null)		92108967R
5 Resis	Diseñar residencias flotantes	01/01/2024	180	15000 (null)		65001721Q

Para el atributo *inv\_principal* hemos seleccionado un profesor para cumplir con las restricciones de integridad; respecto a *empresa*, hablaremos más adelante del diseño empleado. Esta misma metodología se ha utilizado para la **relación GASTO**, donde hemos introducido dos gastos del proyecto de código PESTEL, ambos realizados por su investigador principal.

NUM_G...	CUANTIA	DESC_GASTO	PROYECTO	DNI
1	1	700 Compra de ordenador	PESTEL	47496894W
2	2	20 Licencia de uso de StatGraphics	PESTEL	47496894W



En la **relación PARTICIPA** hemos incluido a todos los becarios (para de esta forma respetar las restricciones de integridad), así como tres profesores para uno de los proyectos:

	DNI	PROYECTO
1	74427528J	BDA
2	84509890H	Fuentes
3	99033805U	PESTEL
4	57039715C	Botellas
5	21833542X	Resis
6	96822803X	BDA
7	22373943S	Fuentes
8	63435235E	PESTEL
9	53949357H	Botellas
10	32772899H	Resis
11	91194873P	BDA
12	63263898F	Fuentes

Por último, las **relaciones de ampliación** también se han cargado manualmente, para ejemplificar modestamente todos los aspectos incluidos en ellas: las subclases PUBLICA y PRIVADA y la relación de conjunto de valores ACCIONISTA\_PRIV. Hemos introducido una empresa pública, ‘Aguas de Valencia’, dependiente del Ayuntamiento, y una empresa privada, ‘Fantastic Computing SL’, que cuenta con dos accionistas. Incluimos capturas, en este orden, de EMPRESA\_EXT, PUBLICA, PRIVADA y ACCIONISTA\_PRIV:

	COD_EMP	NOMBRE_EMP	SECTOR	CONTACTO	PAGO
1	AVLC	Aguas de Valencia SL	Gestión Pública	Joan Ribó	1000
2	FCOMP	Fantastic Computing SL	TIC	Pilar Conesa	20000

	ADMIN	EMPRESA
1	Ajuntament de València	AVLC

	EMPRESA
1	FCOMP

	NOMBRE	EMPRESA
1	Ignacio Gil	FCOMP
2	Pilar Conesa	FCOMP

# Elaboración de informes

Con los datos introducidos, ya podemos preparar nuestros informes para hacer más accesible la información de nuestra base de datos. Más allá de las respectivas explicaciones incluidas a continuación, adjuntamos a este trabajo los seis informes en formato *.rptdesign*.

---

## Informes 1 y 2: Datos de investigadores según su grupo

Tal y como hemos explicado en el apartado anterior, hemos formado dos tipos de grupo: los del sector de Tecnología Virtual y los del sector correspondiente a la Computación Fantástica. Los dos primeros informes reflejan el nombre y el DNI de los investigadores que pertenezcan a dichos sectores, respectivamente.

```
1|select i.nombre_inv nombre, i.dni
2|from investigador i
3|where i.grupo in(select g.nombre_gr
4|                  from grupo g
5|                  where g.nombre_gr = i.grupo and g.desc_gr = 'Tecnología Virtual')
```

### INVESTIGADORES QUE PERTENECEN A GRUPOS DE TECNOLOGÍA VIRTUAL

NOMBRE	DNI
Ricard Standingford	49845540J
Loren Flacke	25417706S
Nowell Borley	49114619T
Katuscha Naish	57082732Q
Shandra Britzius	39091513G
Jorie Grealish	94225914O
Dur Durbin	31783971H
Tiffany Wvch	62053529I

```

1 select i.dni,i.nombre_inv
2 from investigador i
3 where i.grupo in (select g.nombre_gr
4                   from grupo g
5                   where g.nombre_gr=i.grupo and g.desc_gr='Computación Fantástica')

```

## INVESTIGADORES QUE PERTENECEN A GRUPOS DE COMPUTACIÓN FANTÁSTICA

DNI	NOMBRE
01636447K	Clarita Denyer
88965095T	Preston Brandts
60180231F	Hewitt Siaspinski
99992765Q	Darrin Boddy
17899246K	Tina Damsell
63673657S	Glyn Adan
84865737Q	Elane Storev

### Informe 3: Profesores de proyectos independientes

En este caso, reflejamos los datos de los profesores que participan en proyectos que parten del propio departamento; es decir, que no han sido solicitados por empresas externas.

```

1 select nombre_inv nombre, dni
2 from investigador
3 where dni in (select dni from profesor) and dni in (select pa.dni
4                                                    from participa pa
5                                                    where pa.proyecto in (select pr.cod_proy
6                                                            from proyecto pr
7                                                            where pr.empresa is null))

```

## PROFESORES QUE PARTICIPAN EN PROYECTOS INDEPENDIENTES

NOMBRE	DNI
Estrella Giorgione	88698018C
Loren Flacke	25417706S
Gavra Ronaldson	69699460I
Manfred Bugden	92108967R
Ida Gable	76446145R
Jyoti Defries	27839665X
Randie Galletly	65001721Q
Jorie Grealish	94225914O
Kalindi Archibold	89233020E

## Informe 4: Contactos de investigadores principales

Ahora buscamos ofrecer los datos de contacto de todos los investigadores principales de los proyectos del departamento, entre ellos su dirección de correo y sus números de teléfono.

```
1 select t.telefono, i.nombre_inv nombre, i.dni, i.email, g.desc_gr, pa.proyecto
2 from ((investigador i left join grupo g on i.grupo = g.nombre_gr left join telefono_inv t on i.dni=t.dni) left join participa pa on i.dni = pa.dni)
3 where pa.proyecto in (select pr.cod_proy from proyecto pr where pr.inv_principal = pa.dni)
4
5 order by i.dni
```

### CONTACTOS DE INVESTIGADORES PRINCIPALES

TELÉFONO	NOMBRE	DNI	EMAIL	SECTOR	PROYECTO
648435563	Loren Flacke	25417706S	lflacke12@nsw.gov.au	Tecnología Virtual	BDA
628818314	Loren Flacke	25417706S	lflacke12@nsw.gov.au	Tecnología Virtual	BDA
645015300	Simonne Boaler	47496894W	sboaler14@nydailynews.com	Computación Fantástica	PESTEL
689005403	Simonne Boaler	47496894W	sboaler14@nydailynews.com	Computación Fantástica	PESTEL
666639060	Randie Galletly	65001721O	rgalletly13@java.com	Computación	Resis

## Informe 5: Becarios de ministerio

En este informe ofrecemos los datos de todos los becarios seleccionados desde el ministerio, entre ellos el nombre de sus respectivos tutores. La clave para llevar a cabo este informe consiste en concatenar todas las relaciones mediante la instrucción JOIN.

```
1 select i.dni,i.nombre_inv nombre,i.email,i.grupo,pa.proyecto,(select i2.nombre_inv from investigador i2 where i2.dni=b.tutor)tutor ,b.incorporacion
2 from ((investigador i left join bec_ministerio b on i.dni=b.dni) left join grupo g on i.grupo = g.desc_gr) left join participa pa on i.dni = pa.dni
3 where b.tutor is not null
```

### BECARIOS DE MINISTERIO

DNI	NOMBRE	EMAIL	GRUPO	PROYECTO	TUTOR	REGISTRO
35444246V	Jock Degli Antoni	jdegli28@chronoengine.com	TV5	BDA	Renie Ramberg	23 oct. 2023 12:04
58991709K	Duky Delgado	ddelgado20@state.tx.us	TV5	PESTEL	Rheba Stoyles	17 sept. 2023 7:07
92077459T	Lindi Flips	lflips2c@dyndns.org	TV5	Resis	Goldy Quant	13 sept. 2023 19:54
42484516S	Orly Crosseland	ocrosseland24@go.com	TV4	Fuentes	Jelene Chander	11 jul. 2023 17:04
83538312K	Sophia Noyes	snoyes22@dailymotion.com	TV3	Resis	Drona Brito	5 feb. 2023 16:30
17899246K	Tina Damsell	tdamsell1v@e-recht24.de	CE2	BDA	Key Lawson	6 dic 2023

## Informe 6: Investigadores por grupo

Este informe nos indica el número de profesores y de becarios presentes en cada grupo. Cabe destacar que el programa únicamente muestra por pantalla los grupos para los que hay *algún* miembro que sea profesor/becario.

```
1 select g.nombre_gr, count(i.dni)
2 from grupo g, investigador i
3 where g.nombre_gr = i.grupo and i.dni in (select p.dni
4                                           from profesor p
5                                           where p.dni=i.dni)
6 group by g.nombre_gr
7 order by g.nombre_gr
```

```
1 select g.nombre_gr, count(i.dni) NUM_BEC_PROY
2 from grupo g, investigador i
3 where g.nombre_gr = i.grupo and i.dni in (select be.dni
4                                           from bec_proyecto be
5                                           where be.dni=i.dni)
6 group by g.nombre_gr
7 order by g.nombre_gr
```

```
1 select g.nombre_gr, count(i.dni) NUM_BEC_MIN
2 from grupo g, investigador i
3 where g.nombre_gr = i.grupo and i.dni in (select be.dni
4                                           from bec_ministerio be
5                                           where be.dni=i.dni)
6 group by g.nombre_gr
7 order by g.nombre_gr
```

## INVESTIGADORES POR GRUPO

GRUPO	PROFESORES	GRUPO	B. PROYECTO	GRUPO	B. MINISTERIO
CF1	2	CF1	1	CF1	1
CF2	5	CF3	4	CF2	3
CF3	10	CF4	1	CF3	1
CF4	6	CF5	2	CF5	2
CF5	9	TV1	1	TV1	2
TV1	9	TV2	2	TV2	1
TV2	8	TV3	2	TV3	1
TV3	8	TV4	1	TV4	1
TV4	6	TV5	1	TV5	3
TV5	7				

# Programación de módulos

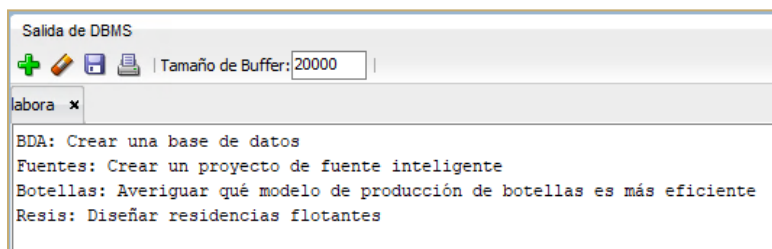
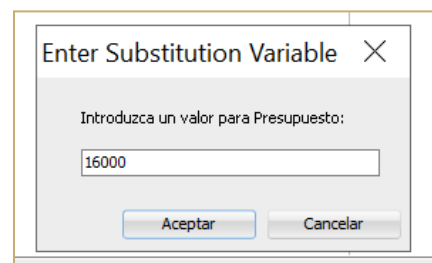
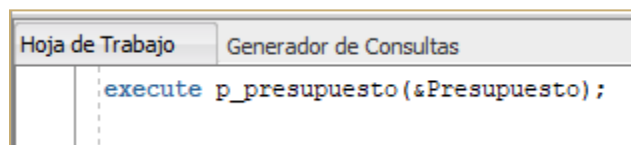
Para concluir nuestro proyecto, hemos programado diversos módulos que nos permiten visualizar mejor los datos de nuestra base, por ejemplo, con solicitudes por parte de usuarios externos en las que, tras introducir un parámetro, la base nos muestre por pantalla la respuesta que satisfaga dicha solicitud.

## Procedimiento 1: Límite de presupuesto

Este procedimiento nos pide que introduzcamos un límite presupuestario (en euros), y nos muestra por pantalla aquellos proyectos con un presupuesto asignado igual o menor.

```
create or replace procedure
p_presupuesto(p_pres proyecto.presupuesto%type)
IS
  cursor proyectos is select cod_proy,desc_proy
  from proyecto
  where presupuesto<=p_pres;
BEGIN
  for v_proy in proyectos loop
    dbms_output.put_line(to_char(v_proy.cod_proy) || ': '
    || v_proy.desc_proy);
  end loop;
end;
```

Aquí mostramos un ejemplo de su ejecución:



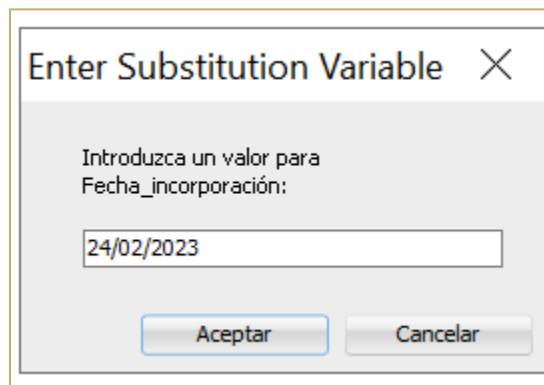
## Procedimiento 2: Filtro de becarios de ministerio por antigüedad

Este procedimiento muestra por pantalla el nombre de todos los becarios de ministerio que se hayan incorporado al Departamento antes de la fecha establecida como parámetro.

```
create or replace procedure
  p_becarios_incorporaron_antes(p_incorporacion bec_ministerio.incorporacion%type)
IS
BEGIN
  for v_bec in (select b.dni,i.nombre_inv,b.incorporacion
  from bec_ministerio b, investigador i
  where b.dni=i.dni and b.incorporacion<p_incorporacion)
  loop
    dbms_output.put_line(v_bec.nombre_inv ||
    ' (' || to_char(v_bec.dni) ||
    '), se incorporó el día ' || v_bec.incorporacion);
  end loop;
end;
```

Así sería la ejecución de este procedimiento:

```
execute p_becarios_incorporaron_antes('&Fecha_incorporación');
```



```
Sophia Noyes (83538312K), se incorporó el día 05/02/2023
Licha Toppes (28930832C), se incorporó el día 21/02/2023
```

### Procedimiento 3: Integrantes de un grupo

Este procedimiento muestra por pantalla el nombre de todos los investigadores que pertenezcan al grupo que hayamos introducido.

```
create or replace procedure
  investigadores_grupo(p_gru grupo.nombre_gr&type)
is
  cursor cur_inv is select nombre_inv
                    from investigador
                    where grupo = p_gru;
  v_cur_inv cur_inv%rowtype;
begin
  open cur_inv;
  loop
    fetch cur_inv into v_cur_inv;
    exit when cur_inv%notfound;
    dbms_output.put_line(to_char(v_cur_inv.nombre_inv));
  end loop;
  close cur_inv;
end;
```

Vemos ahora cómo sería su ejecución:

```
execute investigadores_grupo('&grupo');
```

Enter Substitution Variable X

Introduzca un valor para grupo:

CF1

Aceptar Cancelar

Clarita Denyer  
Darrin Boddy  
Hewitt Siaspinski  
Preston Brandts



## Disparador 1: Grupo inexistente

Este procedimiento nos permite lanzar un error en el caso de que se introduzca un investigador (en la relación INVESTIGADOR) cuyo grupo no forme parte de la relación GRUPO; es decir, controla que en la inserción y/o modificación del grupo de un investigador dicho valor exista en la relación GRUPO.

```
create or replace trigger ins_inv
before insert or update of grupo on investigador
for each row
declare
  v_grupo investigador.grupo%type;
begin
  select nombre_gr
  into v_grupo
  from grupo
  where nombre_gr= :NEW.grupo;
exception
  when no_data_found then
    raise_application_error(-20100,'Error clave ajena: grupo
inexistente');
end;
```

---

## Disparador 2: Bloqueo de grupos

El siguiente disparador prohíbe el borrado y/o modificación de un grupo (relación GRUPO) cuando a dicho grupo pertenece algún investigador.

```
create or replace trigger gru_borrar
before delete or update of nombre_gr on grupo
for each row
declare
  v_inv number;
  hay_investigadores exception;
begin
  select count(*)
  into v_inv
  from investigador i
  where i.grupo = :OLD.nombre_gr;
  if v_inv>0
  then raise hay_investigadores;
  end if;
exception
  when hay_investigadores then
    raise_application_error(-20101,'El departamento tiene investigadores');
end;
```

### Disparador 3: Bloqueo de tutores

El siguiente disparador funciona de manera análoga al anterior, y prohíbe el borrado y/o modificación de un profesor (relación PROFESOR) cuando dicho profesor es tutor de algún becario de ministerio.

```
create or replace trigger gru_borrar
before delete or update of nombre_gr on grupo
for each row
declare
  v_inv number;
  hay_investigadores exception;
begin
  select count(*)
  into v_inv
  from investigador i
  where i.grupo = :OLD.nombre_gr;
  if v_inv>0
  then raise hay_investigadores;
  end if;
exception
when hay_investigadores then
  raise_application_error(-20101,'El departamento tiene investigadores');
end;
```

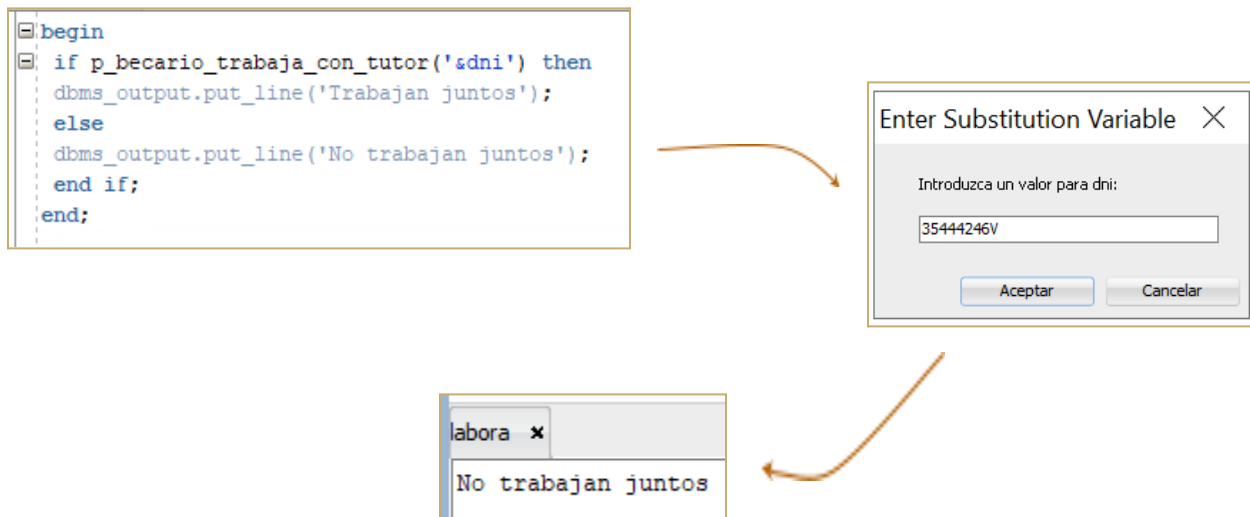
---

### Función 1: Conocer si un becario y su tutor trabajan juntos

En esta función introducimos el DNI de un becario de ministerio, y nos devolverá *True* si su tutor y él participan en el mismo proyecto.

```
create or replace function p_becario_trabaja_con_tutor(
  p_bec_min_dni bec_ministerio.dni%type)
return boolean
is
  v_existe number;
  v_bec_min_dni bec_ministerio.dni%type;
  v_bec_min_tutor bec_ministerio.tutor%type;
begin
  select dni,tutor
  into v_bec_min_dni,v_bec_min_tutor
```

Este sería un ejemplo de la ejecución de la función anterior:



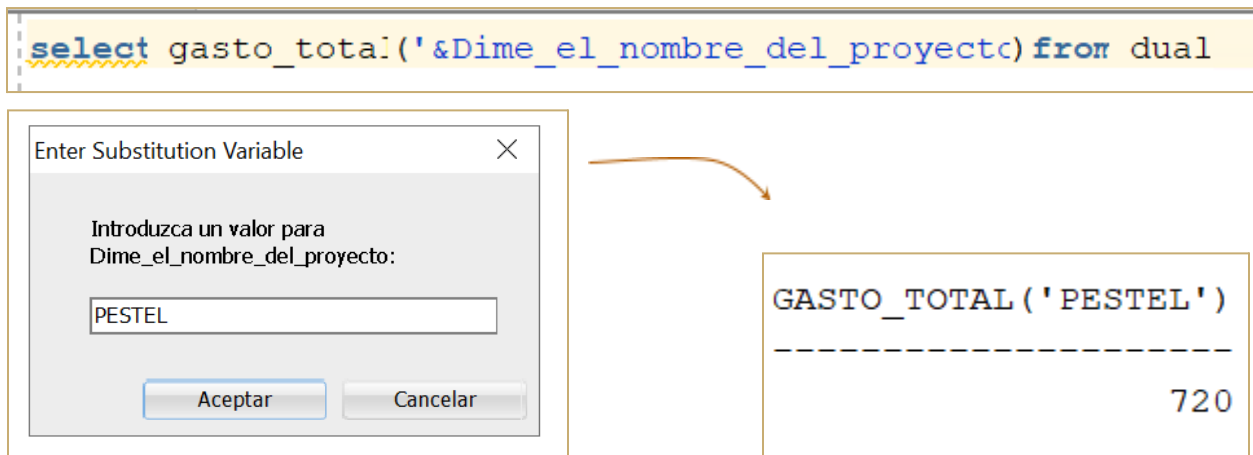
## Función 2: Conocer el gasto total de un proyecto

En esta función introducimos el código de un proyecto, y nos aparece por pantalla el gasto total realizado por los investigadores de dicho proyecto.

```
create or replace function gasto_total(p_proyecto proyecto.cod_proy%type)
return number
is
  v_gasto gasto.cuantia%type;
  v_proyecto proyecto.cod_proy%type;
begin
  select cod_proy
  into v_proyecto
  from proyecto
  where cod_proy = p_proyecto;
  select sum(cuantia)
  into v_gasto
  from gasto
  where proyecto = v_proyecto;
  return v_gasto;
exception
  when no_data_found then
    raise_application_error(-20202, 'Proyecto inexistente!');
end;
```

Ahora vemos un ejemplo de ejecución de la función anterior:

```
select gasto_total('&Dime_el_nombre_del_proyecto') from dual
```



The image shows a database execution environment. At the top, a SQL query is entered: `select gasto_total('&Dime_el_nombre_del_proyecto') from dual`. Below the query, a dialog box titled "Enter Substitution Variable" is displayed. The dialog contains the text "Introduzca un valor para Dime\_el\_nombre\_del\_proyecto:" and a text input field containing the value "PESTEL". There are two buttons at the bottom: "Aceptar" (Accept) and "Cancelar" (Cancel). An arrow points from the dialog box to the right, where the result of the query is shown. The result is a table with one row and one column. The column header is `GASTO_TOTAL('PESTEL')` and the value in the row is `720`.

GASTO_TOTAL('PESTEL')
720

### Función 3: Conocer si un grupo trabaja en un proyecto

Esta función comprueba, dados un grupo y un proyecto, si existen investigadores de dicho grupo que participen en el proyecto especificado.

```
create or replace function comprueba_proyecto(p_grupo grupo.nombre_gr$type,p_proyecto proyecto.cod_proy$type)
return boolean
is
    v_existe number;
    v_nombre_gr grupo.nombre_gr$type;
begin
    select g.nombre_gr
    into v_nombre_gr
    from grupo g
    where p_grupo=g.nombre_gr;

    select count(*)
    into v_existe
    from proyecto p
    where p_grupo in(select grupo from investigador i
                     where i.dni in(select pa.dni from participa pa where pa.proyecto=p.cod_proy)) and p.cod_proy=p_proyecto;
    return((v_existe>0));
exception
    when no_data_found then
        raise_application_error(-20202, 'Grupo inexistente');
end;
```

Este sería un ejemplo de ejecución de la función anterior:

```
begin
if comprueba_proyecto('&grupo','&proyecto') then
  dbms_output.put_line('Hay alguien del grupo trabajando en el proyecto');
else
  dbms_output.put_line('No hay nadie del grupo trabajando en el proyecto');
end if;
end;
```

Enter Substitution Variable X

Introduzca un valor para grupo:

Aceptar Cancelar

Enter Substitution Variable X

Introduzca un valor para proyecto:

Aceptar Cancelar

labora x

Hay alguien del grupo trabajando en el proyecto