
Actividad 11. AEC: Caso práctico IV: Consultas SQL

BASES DE DATOS

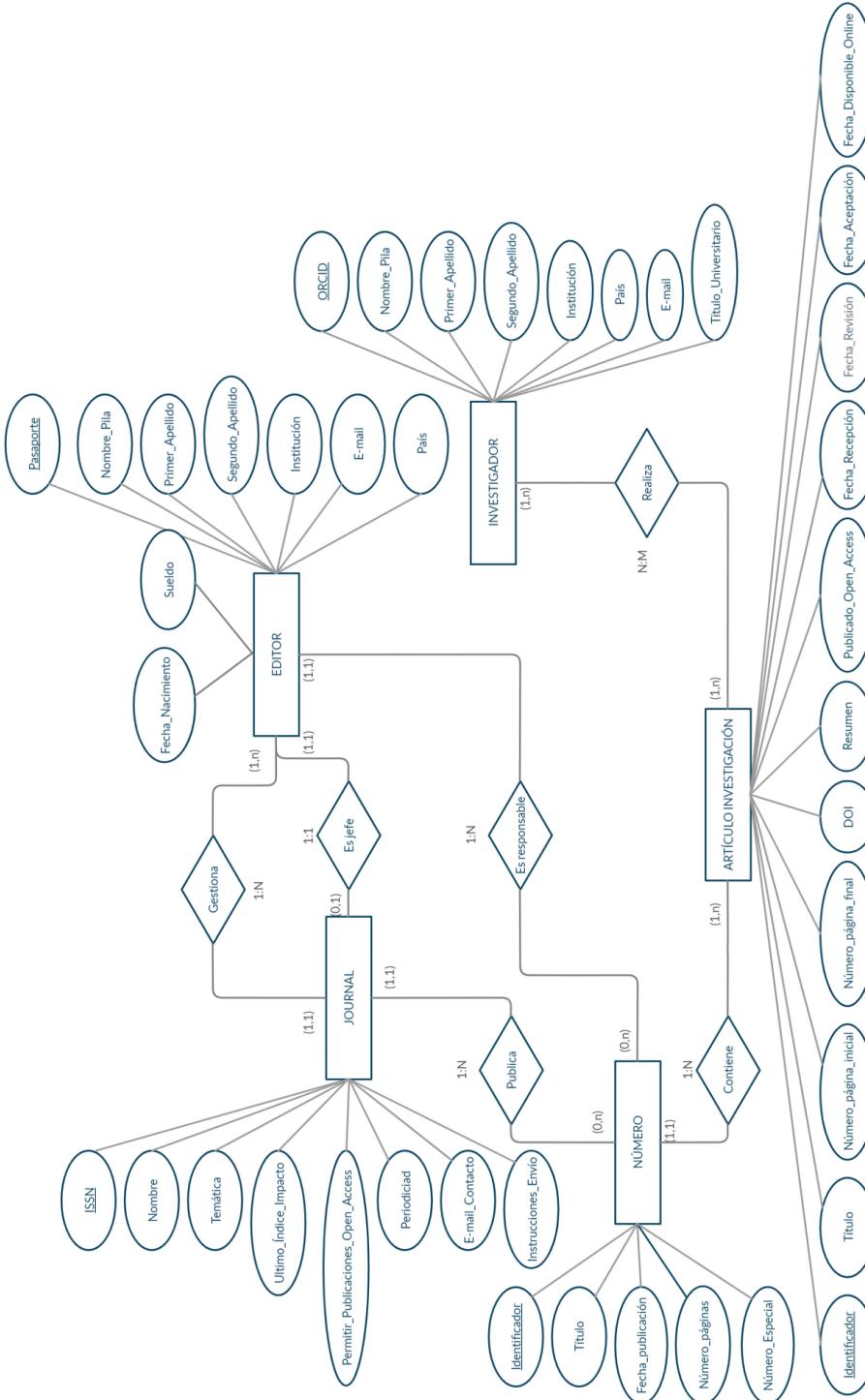


Líder: Alexander Sebastian Kalis
José María Quintanilla Alonso
Raúl Alonso Crespo
Ingeniería de Organización Industrial
1 de enero de 2020

Índice

1. Modelo ER base corregido (CP I)	2
2. Modelo lógico corregido (CP II)	3
3. Correcciones de inserts y scripts (CP III)	4
3.1. script.sql	4
3.2. Inserts.sql	5
4. Consultas SQL (CP IV)	5
4.1. Consulta 1	5
4.2. Consulta 2	6
4.3. Consulta 3	6
4.4. Consulta 4	7
4.5. Consulta 5	8
4.6. Consulta 6	9
4.7. Consulta 7	10

1. Modelo ER base corregido (CP I)



2. Modelo lógico corregido (CP II)

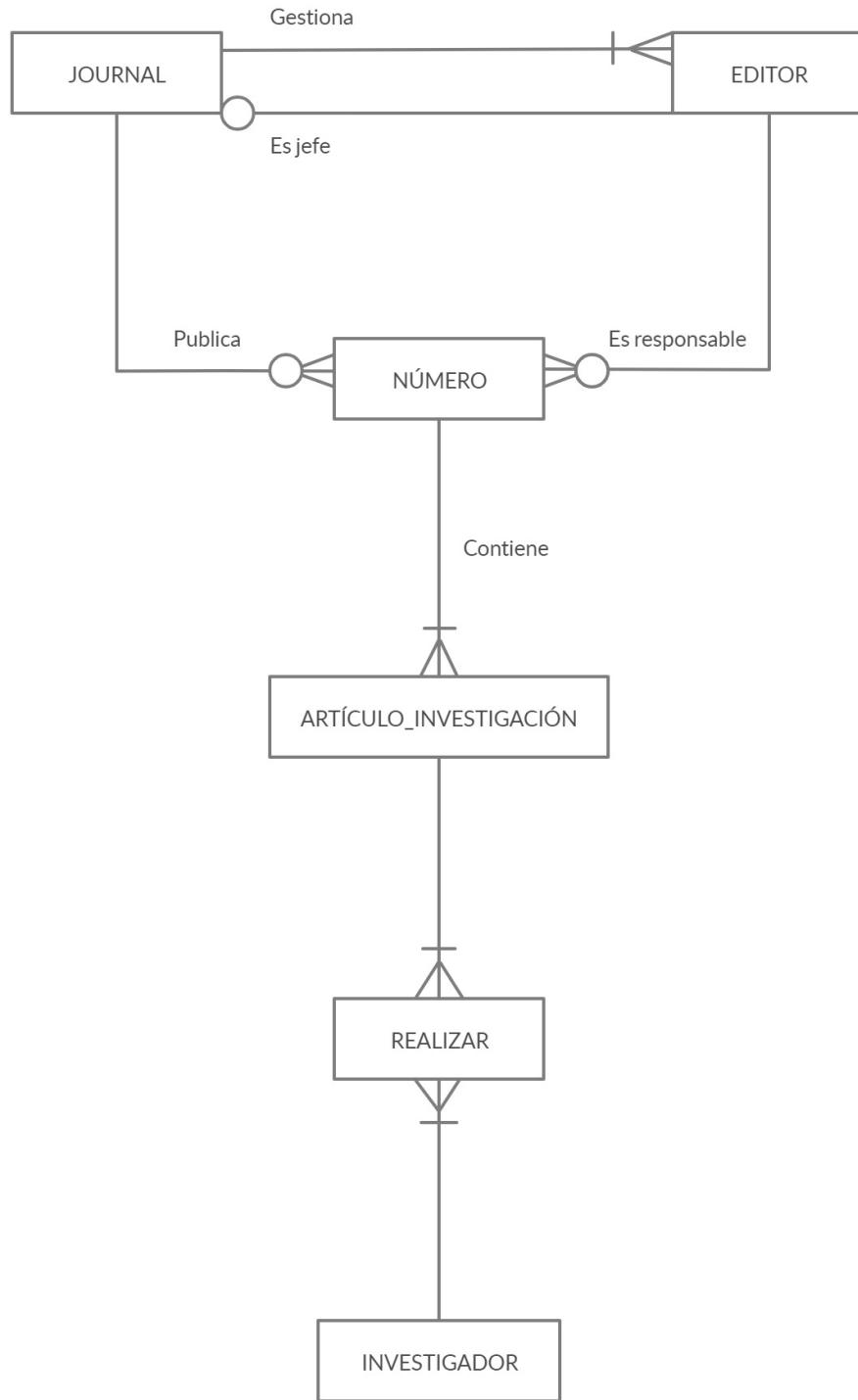


Figura 1: Diseño lógico final

3. Correcciones de inserts y scripts (CP III)

3.1. script.sql

En la creación de la tabla ‘journal’ se ha cambiado el tipo de atributo de ‘Ultimo_Indice_Impacto’ de INT a FLOAT para permitir valores decimales y se ha ampliado el campo de caracteres disponibles a 3000 del atributo ‘Instrucciones_Envio’.

```

• CREATE TABLE IF NOT EXISTS `elsevier`.`JOURNAL` (
    `ISSN` VARCHAR(9) NOT NULL ,
    `Nombre` VARCHAR(40) NOT NULL ,
    `Tematica` VARCHAR(40) NOT NULL ,
    `Ultimo_Indice_Impacto` FLOAT NULL DEFAULT NULL,
    `Periodicidad` VARCHAR(20) NOT NULL ,
    `Permitir_Publicaciones_Open_Access` BOOL NOT NULL,
    `Email_Contacto` VARCHAR(40) NOT NULL ,
    `Instrucciones_Envio` VARCHAR(3000) NOT NULL ,
    `Pasaporte_Editor_Jefe` VARCHAR(12) DEFAULT NULL ,
    -- LA FK Pasaporte editor jefe se anade despues con un alter table --
    PRIMARY KEY (`ISSN`)
);

```

Figura 2: Correciones en tabla journal

En la creación de la tabla ‘articulo_investigacion’, se ha ampliado los caracteres disponibles del atributo ‘Resumen’.

```

• CREATE TABLE IF NOT EXISTS `elsevier`.`ARTICULO_INVESTIGACION` (
    `Identificador` VARCHAR(10) NOT NULL ,
    `Titulo` VARCHAR(180) NOT NULL ,
    `Numero_Pagina_Inicial` INT NOT NULL ,
    `Numero_Pagina_Final` INT NOT NULL ,
    `DOI` VARCHAR(50) NOT NULL ,
    `Resumen` VARCHAR(1000) DEFAULT NULL ,
    `Publicado_Open_Access` BOOL NOT NULL,
    `Fecha_Repcion` DATE NOT NULL ,
    `Fecha_Revision` DATE NULL DEFAULT NULL ,
    `Fecha_Aceptacion` DATE NULL DEFAULT NULL ,
    `Fecha_Disponible_Online` DATE NULL DEFAULT NULL ,
    `Identificador_Numero_Contiene` VARCHAR(10) DEFAULT NULL ,
    PRIMARY KEY (`Identificador`),
    FOREIGN KEY (`Identificador_Numero_Contiene`)
        REFERENCES `elsevier`.`NUMERO` (`Identificador` )
);

```

Figura 3: Correciones en tabla artículo investigación

3.2. Inserts.sql

Debido al cambio de la tabla ‘journal’, se han cambiado también los datos introducidos en la columna ‘Ultimo_Indice_Impacto’ a valores decimales.

```
-- REGISTROS EJEMPLO DE JOURNALS, INDICAMOS TODOS LOS ATRIBUTOS EXCEPTO EL PASAPORTE DEL EDITOR JEFE
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('2049-3630','Immunology','Salud',[2.243],TRUE,'immunology@science.com','200 New York Ave NW Washington DC 20005');
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('2043-5839','Robotics','Ingenieria',[2.532],TRUE,'robotics@science.com','200 New York Ave NW Washington DC 20005');
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('5088-1245','Information Sciences','Ciencia',[5.524], 'Bimensual',TRUE,'infosciences@science.com','200 New York Ave NW Washington DC 20005');
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('5245-3249','Logistics Engineering','Ingenieria Logistica',[1.524], 'Anual',FALSE,'logisticseng@science.com','200 New York Ave NW Washington DC 20005');
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('7334-1889','Chemical Engineering','Ingenieria Quimica',[11.954], 'Semestral',TRUE,'chemicaleng@science.com','200 New York Ave NW Washington DC 20005');
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('1889-2504','Civil Engineering','Ingenieria Civil',[2.772], 'Trimestral',FALSE,'civileng@science.com','200 New York Ave NW Washington DC 20005');
INSERT INTO JOURNAL (ISSN,Nombre,Tematica,Ultimo_Indice_Impacto,Periodicidad,Permitir_Publicaciones_Open_Access,Email_Contacto,Instrucciones_Envio)
VALUES ('3784-4472','Computer Engineering','Ingenieria Informatica',[4.875], 'Mensual',TRUE,'computereng@science.com','200 New York Ave NW Washington DC 20005');
```

Figura 4: Correciones en la inserción de datos de journal

También se han realizado pequeñas modificaciones y adiciones en los valores de los atributos para hacer que los resultados de las consultas muestren al menos un resultado que coincida con todas las condiciones requeridas.

4. Consultas SQL (CP IV)

4.1. Consulta 1

Listado de editores de la editorial, donde aparecerá el nombre del journal en el que presta sus servicios.

Primo apellido	Segundo apellido	Nombre Editor	Nombre journal gestionado
Martel	Lederc	Denis	Civil Engineering
Roberts	Evans	Olivia	Civil Engineering
Medrano	Molina	Ion	Robotics
P.	McIver	Sarah	Immunology
Wilson	Brown	Emily	Computer Engineering
Cock	Dukes	Diane	Computer Engineering
Ortiz	Molina	Pelayo	Information Sciences
McGregor	Molina	Anne	Logistics Engineering
Smith	Jones	Jack	Chemical Engineering
Dubois	Lederc	Dominique	Chemical Engineering

Figura 5: Consulta 1

Descripción

En este caso seleccionamos el nombre completo de la tabla editor y el nombre del journal. Realizamos una unión con la tabla journal bajo la condición de que coincida el ISSN del journal gestionado del editor con el ISSN del journal, para que no aparezcan otras combinaciones editor-journal distintas.

4.2. Consulta 2

Listado de los journals de la editorial, ordenados por índice de impacto (aparecerán primero los de mayor índice de impacto).

The screenshot shows the MySQL Workbench interface with the query editor tab open. The query is:

```

1 -- Consulta 2
2 -- Listado de journals ordenados por su índice de impacto de mayor a menor
3 • USE `elsevier`;
4 • select journal.Nombre as 'Nombre Journal', journal.Ultimo_Indice_Impacto as 'Índice Impacto'
5 from journal
6 order by Ultimo_Indice_Impacto desc

```

The results grid displays the following data:

Nombre Journal	Índice Impacto
Information Sciences	5.524
Computer Engineering	4.875
Civil Engineering	2.772
Robotics	2.532
Immunology	2.243
Chemical Engineering	1.954
Logistics Engineering	1.524

Figura 6: Consulta 2

Descripción

En esta consulta seleccionamos el nombre y el índice de impacto de la tabla journal. Finalmente los ordenamos por el atributo índice impacto de mayor a menor añadiendo el keyword ‘desc’.

4.3. Consulta 3

Listado de los journals de la editorial, junto con el número (cantidad) de editores que gestionan cada uno. Se ordenarán los journals de menor a mayor número de editores.

The screenshot shows the MySQL Workbench interface with the query editor tab open. The query is:

```

1 -- Consulta 3
2 /* Listado de journals con el numero de editores que gestionan cada
3 uno de menor a mayor. */
4 • USE `elsevier`;
5 • select journal.Nombre, count(editor.ISSN_Journal_Gestionado) as 'Gestores'
6 from journal
7 inner join editor
8 on journal.ISSN = editor.ISSN_Journal_Gestionado
9 group by journal.Nombre
10 order by Gestores asc

```

The results grid displays the following data:

Nombre	Gestores
Robotics	1
Immunology	1
Information Sciences	1
Logistics Engineering	1
Civil Engineering	2
Computer Engineering	2
Chemical Engineering	2

Figura 7: Consulta 3

Descripción

Empezamos seleccionando el nombre de la tabla journal y con la función count() el recuento de las filas que contienen valor en el atributo editor.ISSN_Journal_Gestionado. Este recuento lo guardamos bajo alias ‘Gestores’.

Para que ese recuento sea posible, unimos la tabla journal con editor con la condición de relación de ISSN mútuo.

Agrupamos los resultados de conteo de gestores por nombre de cada journal y finalmente utilizamos los resultados guardados en el alias ‘Gestores’ para ordenar de menor a mayor con el uso del keyword ‘asc’.

4.4. Consulta 4

Listado de los investigadores que hayan realizado algún artículo de más de 35 páginas publicado en algún número del journal “Information Sciences”, y cuyo editor responsable (del número) pertenezca a la misma institución que el investigador.

```

37      -- -----
38      -- Consulta 4
39      --
40      /* Listado de los investigadores que hayan realizado algún artículo de más de 35 páginas publicado en algún número del journal
41      "Information Sciences", y cuyo editor responsable (del número) pertenezca a la misma institución que el investigador. */
42      USE 'elsevier';
43      select distinct investigador.Primer_Apellido as 'Primer apellido', investigador.Segundo_Apellido as 'Segundo apellido', investigador.Nombre_Pila as 'Nombre'
44      from investigador
45
46      inner join realizar
47          on investigador.ORCID = realizar.ORCID_Investigador_Realiza
48      inner join articulo_investigacion
49          on realizar.Identificador_Articulo_Realizado = articulo_investigacion.Identificador
50      inner join editor
51          on editor.Institucion = investigador.Institucion
52      inner join numero
53          on numero.Pasaporte_Editor_Responsable = editor.Pasaporte
54
55      where
56      articulo_investigacion.Numero_Página_Final - articulo_investigacion.Numero_Página_Inicial > 35 AND
57      numero.ISSN_Journal_Publicado = '5088-1245';

```

Primer apellido	Segundo apellido	Nombre
Dickinson	Smith	Andrew
South	Arrow	Berta
Jones	Miller	Ava

Figura 8: Consulta 4

Descripción

En esta consulta algo más compleja, seleccionamos (distintos) los nombres completos de los investigadores.

Empezamos realizando las uniones de las tablas que nos van a ser necesarias para aplicar todos los filtros. Se puede observar que se crean dos bloques. El primer bloque son las uniones que tienen como condición relaciones de mutualidad. El segundo bloque son los filtros específicos de atributos de esas tablas.

En el bloque de uniones JOIN, aplicamos directamente el filtro que condiciona que solo deben aparecer los investigadores cuyo artículo pertenezca a un journal que tiene como editor responsable a un miembro de su institución.

En el bloque de condiciones WHERE, realizamos una resta de pagina final e inicial para obtener el número de páginas del artículo y condicionamos que solo aparezcan los investigadores cuyo artículo supera las 35 páginas.

Por último condicionamos que sólo aparezcan los nombres de investigadores cuyo artículo sea publicado en el journal con ISSN ‘5088-1245’, que corresponde al journal “Information Sciences”.

4.5. Consulta 5

Listado de artículos open access con 4 o más autores, contenidos en números de journals con periodicidad trimestral gestionados por menos de 10 editores.



```

consulta5* | consulta6  consulta7  articulo_investigacion  numero
1   -- Consulta 5
2   /*
3     Listado de artículos open access con 4 o más autores, contenidos en números de
4     journals con periodicidad trimestral gestionados por menos de 10 editores.
5   */
6 • USE `elsevier`;
7 • select articulo_investigacion.Titulo
8   from articulo_investigacion
9
10  join numero
11    on articulo_investigacion.Identificador_Numero_Contiene = numero.Identificador
12  join journal
13    on numero.ISSN_Journal_Publicado = journal.ISSN
14
15  where
16    articulo_investigacion.Publicado_Open_Access = true AND
17    journal.Periodicidad = 'Trimestral' AND
18
19    -- Subquery, condicion de 4 o mas autores
20    articulo_investigacion.Identificador IN
21  (
22    select realizar.Identificador_Articulo_Realizado
23      from realizar
24      group by realizar.Identificador_Articulo_Realizado
25      having count(Identificador_Articulo_Realizado) >= 4
26  )
27  AND
28    -- Subquery, condicion de menos de 10 editores
29    journal.ISSN IN
30  (
31    select editor.ISSN_Journal_Gestionado
32      from editor
33      group by editor.ISSN_Journal_Gestionado
34      having count(editor.ISSN_Journal_Gestionado) < 10
35  )

```

Figura 9: Consulta 5

Descripción

Similarmente a la consulta 4, empezamos por unir tablas con relación de mutualidad.

Posteriormente aplicamos filtros condicionales donde especificamos que el artículo debe ser open access y que debe estar en un journal con periodicidad trimestral.

Para mostrar solamente aquellos artículos que tengan 4 o más editores, se ha creado una subconsulta la cual nos devuelve una lista de IDs de artículos;

Esta sublista está condicionada con el uso del keyword ‘having’, haciendo que sólo se muestre en la sublista aquellas IDs (agrupadas por ID) de artículos que se repiten 4 o más veces (utilizando la función count()) en la tabla ‘realizar’ ya que eso significa que tienen 4 o más autores. Por último, mediante el keyword ‘IN’, obligamos a que la consulta principal nos muestre solamente los artículos cuyo ID aparezca en esta sublista.

De la misma forma se ha realizado la condición de que solamente se muestren los artículos cuyo journal tiene menos de 10 editores.

4.6. Consulta 6

Sueldo medio de los editores, jefes de algún journal con algún número especial publicado en 2018.

```

consulta1 consulta2 consulta3 consulta4 consulta5 consulta6 consulta7
1 -- Consulta 6
2 /*
3 Sueldo medio de los editores, jefes de algún journal con algún número especial publicado en 2018.
4 */
5 • USE `elsevier`;
6 • select AVG(editor.Sueldo) as 'Sueldo Medio'
7 from editor
8 where
9 -- Subquery, editores que son jefe de un journal
10 editor.Pasaporte IN
11 (
12     select journal.Pasaporte_Editor_Jefe
13     from journal
14     where
15         -- Subsubquery, journals que contienen un numero especial y que sea publicado en 2018
16         journal.ISSN IN
17             (
18                 select numero.ISSN_Journal_Publicado
19                 from numero
20                 where
21                     numero.Numero_Especial = true AND
22                     numero.Fecha_Publicacion >= '2018-01-01' AND
23                     numero.Fecha_Publicacion < '2019-01-01'
24             )
25 )
26
Result Grid | Filter Rows: | Export: | Wrap Cell Content: |
| Sueldo Medio | 34000.000000 |

```

Figura 10: Consulta 6

Descripción

En esta consulta seleccionamos la media de los sueldos de la tabla editor mediante el uso de la función avg() y lo guardamos bajo alias ‘Sueldo medio’

Para los filtros, hacemos una subconsulta dentro de otra subconsulta:

En la primera subconsulta, especificamos que el pasaporte del editor (que debe ser jefe de un journal) estará contenido en (IN) la lista de pasaportes que son jefes de la tabla journal.

Sin embargo, con el uso de una subconsulta dentro de esa subconsulta, hacemos que sólo aparezcan los pasaportes de journals los cuales contienen algún número que sea de tipo especial y además su fecha de publicación esté comprendida entre 2018-01-01 y 2019-01-01 para que sea del 2018.

4.7. Consulta 7

Número medio de artículos por número (se pide un único valor).

```

129
130      --
131      -- Consulta 7
132      --
133  /*
134   * Número medio de artículos por número (se pide un único valor).
135   */
136 • USE `elsevier`;
137 • select avg(articulosPorNumero) as 'Media articulos por numero'
138   from
139   (
140     select count(articulo_investigacion.Identificador_Numero_Contiene) as articulosPorNumero
141       from articulo_investigacion
142         group by articulo_investigacion.Identificador_Numero_Contiene
143   ) tablaArticulosPorNumero; -- Tabla derivada
144

```

Media articulos por numero
1.3333

Figura 11: Consulta 7

Descripción

En esta consulta se utiliza el concepto de tabla derivada, es decir, una tabla volátil que se inicializa en el runtime de la consulta.

Lo que hacemos es seleccionar la media con `avg()` de los números que contiene cada artículo y se muestra bajo alias ‘Media articulos por numero’.

Para obtener los números por artículo creamos una tabla derivada en la cual seleccionamos el recuento (con `count()`) de los identificadores del número que contiene cada artículo y lo guardamos bajo alias ‘articulosPorNúmero’. Agrupamos por ese identificador para que cada número tenga su cantidad de artículos.

Finalmente el lenguaje nos obliga a guardar esta tabla derivada bajo un nombre. En este caso se ha guardado bajo el nombre ‘tablaArticulosPorNumero’.