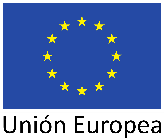




PLAN DE PRUEBAS



Programa Operativo Fondo Europeo de Desarrollo Regional Aragón 2014-2020

*Construyendo Europa desde Aragón*

Contenido

[1. Análisis exploratorio 5](#_Toc59467082)

[1.1 Corpus utilizado 5](#_Toc59467083)

[2. Pruebas de las expresiones regulares 7](#_Toc59467084)

[2.1 Proceso de creación de las expresiones regulares 7](#_Toc59467085)

[3. Etiquetamiento manual 9](#_Toc59467086)

[3.1 Etiquetas y política de etiquetamiento 9](#_Toc59467087)

[3.2 Corpus utilizado 10](#_Toc59467088)

[3.3 Número de ocurrencias etiquetadas 10](#_Toc59467089)

[4. Pruebas para el modelo NER 11](#_Toc59467090)

[5. Pruebas unitarias y de integración 14](#_Toc59467091)

[6. Pruebas de aceptación 15](#_Toc59467092)

[6.1 Caso de prueba de ingesta 15](#_Toc59467093)

[6.1.1 Caso 1 15](#_Toc59467094)

[6.2 Caso de prueba de conversión 16](#_Toc59467095)

[6.2.1 Caso 2 16](#_Toc59467096)

[6.3 Casos de prueba de extracción y almacenamiento 16](#_Toc59467097)

[6.3.1 Caso 3 17](#_Toc59467098)

[6.3.2 Caso 4 18](#_Toc59467099)

[6.3.3 Caso 5 19](#_Toc59467100)

[6.3.4 Caso 6 20](#_Toc59467101)

[6.3.5 Caso 7 21](#_Toc59467102)

[6.3.6 Caso 8 22](#_Toc59467103)

[6.3.7 Caso 9 23](#_Toc59467104)

[6.3.8 Caso 10 24](#_Toc59467105)

[6.3.9 Caso 11 25](#_Toc59467106)

[6.3.10 Caso 12 26](#_Toc59467107)

[6.3.11 Caso 13 27](#_Toc59467108)

[6.3.12 Caso 14 28](#_Toc59467109)

[6.3.13 Caso 15 29](#_Toc59467110)

[6.3.14 Caso 16 30](#_Toc59467111)

[6.3.15 Caso 17 32](#_Toc59467112)

[6.3.16 Caso 18 33](#_Toc59467113)

[6.3.17 Caso 19 34](#_Toc59467114)

[6.3.18 Caso 20 35](#_Toc59467115)

[6.3.19 Caso 21 36](#_Toc59467116)

[6.1 Caso de prueba de cambios en los ficheros de configuración 37](#_Toc59467117)

[6.1.1 Caso 22 37](#_Toc59467118)

[6.2 Casos de pruebas cierre de ofertas 40](#_Toc59467119)

[6.2.1 Caso 23 40](#_Toc59467120)

[6.2.2 Caso 24 42](#_Toc59467121)

[6.2.3 Caso 25 44](#_Toc59467122)

Control de versiones

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Fecha | Autor | Descripción |
| 1.0 |  | Hiberus | Primera versión (incompleta) del documento |

# Análisis exploratorio

En esta sección se va a comentar el corpus utilizado para realizar el análisis exploratorio inicial, que sirve para **estudiar la estructura y el formato de los artículos de los diferentes tipos de boletines**. Cabe destacar que, dado que desarrollando los diferentes pipelines se realizaban pruebas unitarias continuamente, se observaron (y se tuvieron en cuenta) **más documentos de los aquí indicados**. Por ejemplo, en este corpus no aparecen apenas artículos de cierres de ofertas, que se analizaron sin almacenarlos en el corpus.

## Corpus utilizado

A continuación, se va a detallar el corpus utilizado para el análisis exploratorio inicial, cuyo **árbol de directorios** sigue la estructura de la Ilustración 1 (se ha cortado la imagen por efectos prácticos: el Corpus\_XML aparecido abajo sigue la estructura de Corpus\_PDF, y también existe Corpus\_HTML con la misma estructura). Se almacenaron los mismos artículos en sus **tres formatos** (pdf, html y xml). Se usaron, además de artículos con ofertas de empleo (Sueltos/Empleo), artículos de otras temáticas (Sueltos/Otros) para conocer mejor la estructura de todos ellos. Se manejaron **temáticas muy variadas** para este caso, desde artículos sanitarios a presupuestarios, pasando por decretos frente al terrorismo. En Sumarios se guardaron algunos sumarios también con intención de conocer su formato.

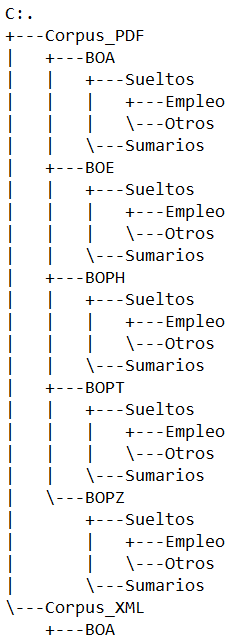


Ilustración 1. Parte del árbol de directorios del corpus de análisis

En cuanto al número de artículos almacenados y estudiados en este análisis:

* Para BOA y BOE, se utilizaron 14 documentos de empleo, 6 de otras temáticas y 5 sumarios.
* Para BOPH, BOPZ y BOPT, 5 documentos de empleo, 3 de otras temáticas y 3 sumarios. Se manejaron menos debido a la dificultad de encontrar documentos de empleo de forma manual.

En cuanto a los artículos de empleo, se realizó una búsqueda concienzuda para encontrar documentos variados en cuanto a estructura, para tratar de abarcar cuanto antes todos los posibles casos con los que se iba a encontrar el código posteriormente. También se tuvo en cuenta el recuperar documentos de a lo largo de todo el año (hasta septiembre, momento del análisis). De esta forma, los artículos manejados con ofertas de empleo son:

* BOA: 1 de enero, 2 de febrero, 2 de marzo, 1 de mayo, 3 de junio, 2 de julio, 2 de agosto y 1 de septiembre.
* BOE: 2 de enero, 3 de marzo, 1 de abril, 1 de mayo, 2 de junio, 1 de julio, 3 de agosto, 1 de septiembre.
* BOPH: 1 de julio, 3 de agosto y 1 de septiembre.
* BOPZ: 1 de enero, 2 de junio, 1 de julio y 1 de septiembre.
* BOPT: 1 de junio, 1 de julio, 1 de agosto y 2 de septiembre.

En el documento de validación del análisis exploratorio se ahonda más en este corpus y en lo visto en él.

# Pruebas de las expresiones regulares

En esta sección se va a comentar el proceso realizado para probar las expresiones regulares, donde se comenta el corpus utilizado inicialmente.

## Proceso de creación de las expresiones regulares

Se almacenaron diferentes documentos en los que se vieron discrepancias en los campos, pero se observaron muchos más: si se observaban campos con las mismas sintaxis, solo se utilizaba uno de ellos como corpus para estas pruebas.

Inicialmente el corpus se creó a partir de una selección realizada con el método que se acaba de comentar sobre el corpus del análisis, guardando los documentos de texto tanto del propio texto como del título.

A partir de estos documentos, se estudió la **sintaxis** de algunos de los campos, así como la ubicación de aparición de otros o incluso el contexto de algunos de ellos. Para algunos campos más extensos se estudiaron las sintaxis con más profundidad, para dar con el patrón que abarque todas las formas observadas. Ejemplos de ello son la Ilustración 2 y la Ilustración 3.

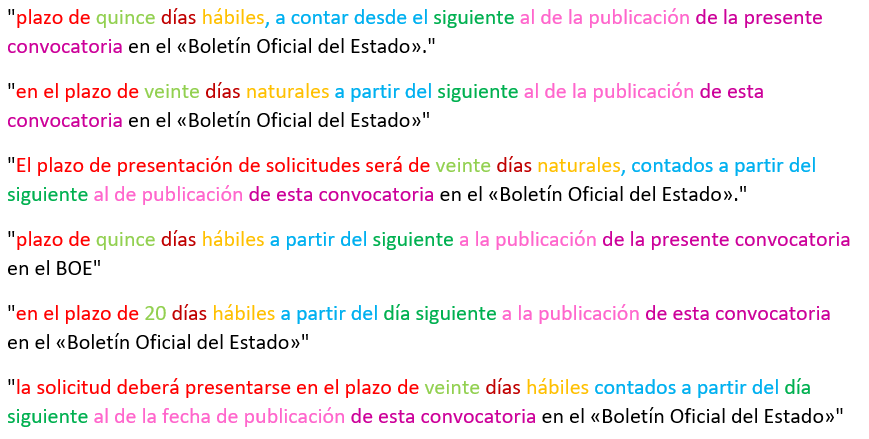


Ilustración 2. Sintaxis del plazo en el BOE

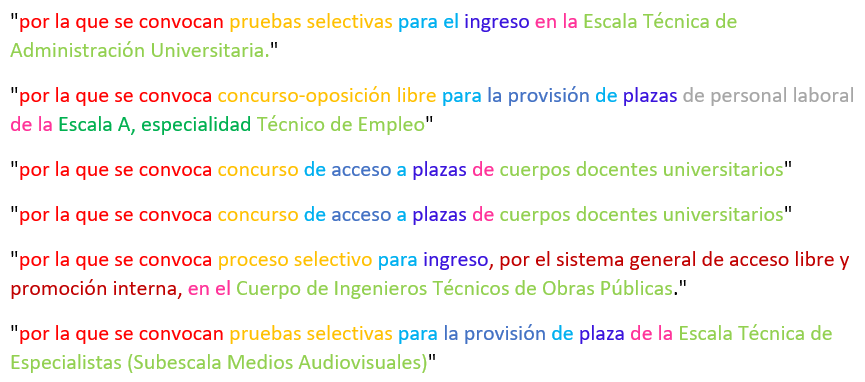


Ilustración 3. Sintaxis de puestos en el título del BOE

A continuación, se crearon unas **expresiones regulares iniciales**, que se adaptaban bien al corpus del momento. Sin embargo, estas expresiones han ido siendo modificadas continuamente conforme aparecían textos de los distintos campos que no se adaptaban completamente a las reglas. Por ello, se modificaba la expresión regular pertinente para la nueva observación, teniendo en cuenta no cambiar nada que se saliera del patrón de las observaciones anteriores.

Además, como en algunas reglas se extrae también algo de **contexto** para asegurar que se obtiene el texto del lugar adecuado, es necesario limpiar (o comprender, en algunos casos) este texto. Por ello, algunas reglas vienen divididas en varias partes, como la del plazo, que ha de entender la información para definir las fechas de presentación de solicitudes.

Tras realizar el proceso comentado las expresiones regulares finalmente son las que aparecen en el fichero de configuración, modificable, **‘regex.xml’**, cuyo significado se detalla en el Manual Técnico.

# Etiquetamiento manual

En esta sección se va a hablar sobre el etiquetamiento manual realizado para poder entrenar con confianza el modelo de reconocimiento de entidades.

## Etiquetas y política de etiquetamiento

Las etiquetas empleadas son las siguientes: puesto, escala, cuerpo, plazo, grupo, tipo\_convocatoria, num\_plazas, web e email (estas dos últimas, pertenecientes a datos\_contacto). El resto de los campos se obtiene mediante los propios metadatos o mediante reglas (algunos de los aquí tratados también).

Para estas reglas se definió una **política de etiquetamiento**, necesaria para seguir siempre los mismos criterios para los textos que aparecían, algo especialmente interesante si etiqueta más de una persona. Esta política de etiquetamiento se centra en casos concretos, abogando al sentido común para casos más generales. El objetivo final es estandarizar los textos etiquetados, para normalizar la salida esperada. Se detalla la política de cada etiqueta:

* **Puesto:** No se incluirán como puesto ni los cuerpos ni las escalas. Se incluirán todos los puestos, aunque los mismos no sean los ofertados en el artículo, dándole así más vocabulario (válido) al modelo. En los casos en los que el puesto venga precedido por una especie de etiquetado (por ejemplo, “Denominación: ”), esa parte también irá incluida en la etiqueta, ya que se eliminará en un procesado posterior. No se etiquetan como puestos los textos parciales que pudieran aparecer de lecturas erróneas de tablas verticales de los anexos.
* **Plazo:** En el caso habitual, se incluirá desde el número de días que indica hasta la palabra “publicación”. Por ejemplo, en el caso “El plazo para la presentación de solicitudes será de veinte días hábiles contados a partir del día siguiente al de la fecha de publicación de la convocatoria en el «Boletín Oficial del Estado»”, solo se considerará plazo lo subrayado. Esto normaliza mucho más los plazos, sin perder información que interese. En los casos extraordinarios, como los que simplemente se indica una fecha de finalización, se tratará de ajustar a las mismas directrices de guardar únicamente lo necesario para su correcta interpretación.
* **Web:** Se guarda la URL completa, aunque en medio de la misma aparezca un espacio. Tratar de evitar puntos y comas tras la misma. Solo se guardan URLs, no indicaciones de ir a páginas web, como “Portal Formul@”.
* **Email:** Se guarda el correo electrónico completo, aunque en medio de este aparezca un espacio. Tratar de evitar puntos y comas tras el mismo. No importa si aparece en mayúsculas o con puntos en el prefijo.
* **Escala:** Se guarda como escala cualquier escala o subescala que aparezcan, siempre y cuando lleven dicho nombre como prefijo, e incluyéndolo en la etiqueta.
* **Cuerpo:** Se guardan como cuerpo los cuerpos que se consideren que tienen una cierta nomenclatura de cuerpo, siempre y cuando lleven cuerpo/cuerpos como prefijo, e incluyéndolo en la etiqueta. Un contraejemplo sería “una reserva para desempeñar puestos como interino en otro Cuerpo docente no universitario”. En dicho caso, no se indica un cuerpo concreto, por lo que no se etiqueta. “Cuerpo de la Policía Local”, por ejemplo, sería un cuerpo correcto.
* **Grupo:** Se guarda el grupo o subgrupo cuando aparece con dicho prefijo, incluyéndolo. No se etiqueta por ejemplo únicamente “C1”, porque puede confundirse con el nivel de inglés. En los casos en los que indique “grupo/subgrupo C1”, todo se guarda como etiqueta. En ocasiones, puede aparecer un espacio, como por ejemplo “subgrupo A 1”. En esas ocasiones, coger todo. A veces aparece como “grupo profesional”. En caso de aparecer varios, con la forma “Subgrupo C1 o A2”, coger todo como etiqueta.
* **Tipo\_convocatoria:** Se etiqueta con alguno de los términos detectados, que aparecen en la validación del análisis exploratorio. Especialmente en el caso de que ambos tipos estén contemplados en la misma convocatoria, etiquetar la línea estandarizada que lo indica. Posteriormente, indicar ambos indistintamente, para que el modelo tenga más repeticiones. En los casos en los que especifique demasiado, como “turno libre ordinario” o “turno libre de reserva”, quedarse con el término más general, en estos casos lo subrayado. No etiquetar únicamente con la palabra “libre”, ya que puede referirse a turno libre o a libre designación, que indican tipos opuestos.
* **Num\_plazas:** En los casos como “15 plazas”, todo es etiqueta. Cuando ponga “el puesto vacante”, solo se omite el artículo, ya que a veces pondrá “del”. Cuando venga “etiquetado”, como por ejemplo “Número de plazas: Una”, todo será etiqueta, procesándolo posteriormente.

Con estas directrices se ha etiquetado el corpus detallado en el apartado siguiente, obteniendo una salida suficientemente normalizada en la mayoría de los casos.

## Corpus utilizado

Primero se etiquetaron los artículos de ofertas de empleo almacenados como corpus de análisis, para luego incorporar muchos más documentos, que se obtenían mediante los pipelines de ingesta y conversión de días elegidos de forma arbitraria, para que hubiera variación de fechas. Además, periódicamente se realizaban copias de seguridad de toda la información etiquetada, para poder recuperar la información en caso de pérdida.

Pertenecen a este corpus (y por consiguiente se han etiquetado) **98 documentos del BOA, 107 documentos del BOE y 23 documentos de boletines provinciales**. Se le dio preferencia a los del BOA, pero es tan numerosa la cantidad de artículos del BOE que se ha tratado una cifra mayor de este. En total se han etiquetado 228 documentos.

## Número de ocurrencias etiquetadas

Tratando de etiquetar el número máximo de textos posible en el corpus, el número de ocurrencias etiquetadas finalmente por etiqueta es el siguiente:

* Puesto: 994
* Plazo: 285
* Web: 186
* Email: 46
* Escala: 278
* Cuerpo: 147
* Grupo: 176
* Tipo\_convocatoria: 690
* Num\_plazas: 390

Con estos números se puede intuir la cantidad de veces que aparece cada campo, pero no su distribución, ya que algunos campos como la escala en la mayoría de documentos no aparece, pero en los que aparece se repite múltiples veces. Algo similar ocurre con el puesto, especialmente cuando aparecen muchos puestos en tablas.

# Pruebas para el modelo NER

En esta sección se van a explicar las pruebas realizadas para la comprobación del correcto funcionamiento del modelo NER.

Para probar el NER de Spacy, primero se realizó una prueba de concepto de la misma, viendo cómo trabajaba con datos y ejemplos predefinidos. Tras esta primera prueba, se procedió a realizar la integración de la herramienta con Doccano, de forma que recuperara las anotaciones directamente de Doccano. Tras estas primeras pruebas, se pasó a etiquetar documentos en Doccano.

Una vez ya se contaba con documentos propios etiquetados, y para empezar a comprobar que el NER detectaba correctamente (al menos los textos con los que se ha entrenado), se han realizado los siguientes experimentos:

Prueba 1

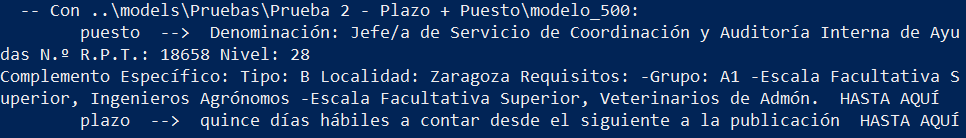
Se ha creado el proyecto “Prueba plazo” en Doccano (proyecto 4) y se le ha añadido únicamente la etiqueta “plazo”. Se ha incorporado como *dataset* el BOA\_20200311\_2 del corpus de modelado y se ha etiquetado la única ocurrencia de plazo que aparece.

Se han creado y entrenado los modelos con este proyecto, con 1, 5, 10, 25, 50, 100 y 500 iteraciones sobre el *dataset*.

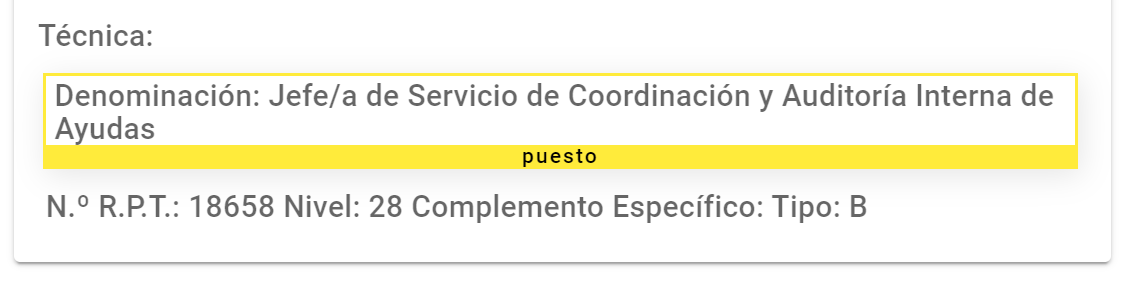
Para terminar esta pequeña prueba, se ha evaluado el mismo documento etiquetado con todos estos modelos. Con 1, 5, 10 y 25 iteraciones, se encontraban las mismas (muchas) etiquetas de las que vienen por defecto (PER, LOC, ORG, MISC). Con 50 encuentra únicamente 5 etiquetas por defecto. Con 100 no encuentra nada (olvida las por defecto por la información propia pasada). Con 500 encuentra, por fin, el plazo etiquetado.

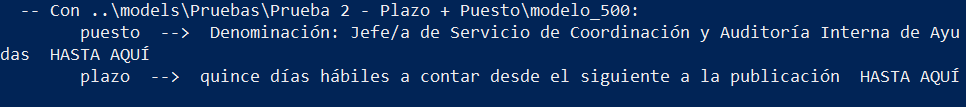
Prueba 2

Sobre el mismo proyecto (“Prueba plazo”), se ha creado la etiqueta “puesto” y se ha etiquetado el mismo. Como se cortaba entre documento y documento, se ha movido el texto justo para que apareciera entero en el mismo documento. Sin embargo, al evaluar el mismo documento, el puesto que encontraba (con 500 iteraciones, ya que ocurre lo mismo que con la Prueba 1), recogía también texto que aparecía tras el mismo.



Volviendo al Doccano y subiendo el texto posterior al documento, ubicándolo tras la etiqueta, ya no ocurre lo mencionado en la evaluación. En un principio esto llevó a pensar que el NER le da mucha importancia al contexto para detectar entidades.





Reentrenando y reevaluando estos modelos sin modificar nada se ha podido comprobar que no devuelve siempre lo mismo. De hecho, en ocasiones devuelve el puesto de forma incorrecta y no devuelve el plazo y otras devuelve ambos bien. En la prueba siguiente se encontró el problema.

Prueba 3

Sobre el mismo proyecto, se pasó a añadir un segundo documento para comprobar qué entidades encuentra con más de un documento. En este momento, y tras un estudio extenso de la herramienta, se encontraron varios problemas que estaban ocurriendo con el modelo:

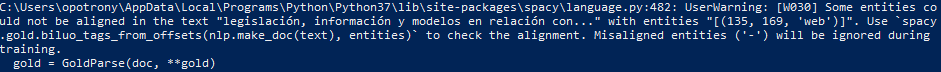
* Para la recuperación de páginas de Doccano existen los parámetros *limit* y *offset*. El primero indica el número de documentos que va a contener la página recuperada como máximo. El segundo, a partir de qué documento se cogen *limit* documentos. (Es decir que, si queremos obtener los documentos por páginas de 10 documentos y, en concreto, la página a recuperar es la primera, los parámetros serán limit=10, offset=0. Para la siguiente página, limit=10, offset=10). El problema es que, inicialmente, se indicaba un limit de 10 y tan solo recuperaba 5 documentos. Investigando los ficheros del repositorio de GitHub de ‘Doccano\_api\_client’ no se encontró la razón para que, poniendo cualquier límite, este recupere únicamente páginas con límite 5. Para solucionarlo, se utilizó de forma propia dicho límite.
* Otro error encontrado, esta vez propio es que, mientras que la función *read\_annotations\_from\_doccano* devuelve las páginas con diferentes instrucciones *yield*, en el script que se la invocaba se pasaba el resultado inmediatamente a una lista, por lo que solo se obtenía el primer *yield,* correspondiente a la primera página (que, como se ha comentado en el punto anterior, eran siempre un máximo de 5 documentos aunque se indicaran 100).
* También se encontró un error con menos consecuencias. También en la función *read\_annotations\_from\_doccano*, se puede observar que se realiza la división entera entre el número de documentos y el límite para obtener el número de páginas a tratar. La división entera es como realizarle un suelo a dicha división por lo que, para 13 documentos y un límite de 10, se obtiene una única página. Esto resulta en la pérdida de los últimos 3 documentos que, cuando se tiene un dataset muy pequeño (como es el caso de esta tercera prueba), pueden ser significativos. Se ha solucionado obteniendo el id del último documento tratado e incrementándolo hasta cubrir el número de documentos restantes, que se obtienen con el método *get\_document\_detail()*.
* Por último, se encontró otro error grave originario del paquete del doccano\_api\_client. Cuando se invoca la función, dentro de *read\_annotations\_from\_doccano()*, *get\_document\_list()*, hay que pasarle como parámetros el id del proyecto y un diccionario con los parámetros *offset* y *limit*, como se ha comentado anteriormente. El problema radica en que, cuando se le pasa como offset un número de más de una cifra, solo utiliza la última cifra (por ejemplo, si se pasa un offset ‘10’ o ‘20’, coge los documentos con offset ‘0’). Esto repercute en que siempre coge los documentos de la primera página. En el repositorio de github se ha estudiado esta función, que trata los parámetros con la función *build\_url\_parameter()*. Estudiando esta función, se ha visto que el constructor de la URL es el punto problemático, ya que las construye creando un offset con cada cifra pasada como offset, y un limit con cada cifra pasada como limit.

Como se ha comentado, varios de los errores que no se habían detectado vienen originados por las propias herramientas, que al ser *open-source* no vienen del todo depuradas.

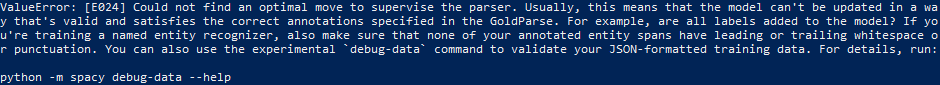
La solución de estos errores conllevó la desaparición de los problemas mencionados en la prueba 2: **el modelo ya trabaja bien resultando en una misma salida para una misma entrada, y los textos que ya ha memorizado los devuelve con la misma salida**. Además, el número de iteraciones necesarias para su correcto funcionamiento bajó bastante: con 50 iteraciones en lugar de 500 ya clasifica satisfactoriamente.

Prueba 4

Utilizando ya el proyecto de Doccano con todas las anotaciones realizadas, se ha comprobado que aparecen una serie de *warnings* al ejecutar el NER, del tipo:



Además, tras una serie de estos *warnings* llegaba el error, que termina el programa ya que no ha conseguido hacer varios *updates* de forma correcta:



Estos *warnings* (y en consecuencia el error) se deben a que las anotaciones que se indican no coinciden con la tokenización del texto. Esto se debe a 3 causas principales: espacios en blanco al principio de la anotación, espacios en blanco al final de la anotación y, sobre todo, que no se coja un token completo para una anotación, sino solo una parte de él. Esto puede deberse a algún fallo en el etiquetado (seleccionar únicamente “bogado” de la palabra “Abogado”) o a no coger un signo de puntuación para la anotación (coger “cuerpo de funcionarios” en el texto “cuerpo de funcionarios.”).

Se han conseguido evitar estos casos limpiando las anotaciones y los textos de forma previa a estos *updates*. Esta limpieza consistiendo en eliminar posibles espacios en blanco previos y posteriores incluidos en la etiqueta y añadir un espacio en blanco anterior y posterior a las etiquetas cuya tokenización se ve partida.

Pruebas posteriores

A partir de este momento, las pruebas realizadas pueden considerarse **unitarias**, ya que fueron pruebas con distintos documentos para comprobar el correcto funcionamiento del modelo, y ver hasta dónde puede llegar el mismo.

Estas pruebas se realizaron en su mayor parte con documentos no utilizados en el etiquetado (ni en el análisis ni en el corpus de las reglas, ya que ambos están contenidos en el corpus de etiquetado), procediendo estos documentos de ingestas y conversiones de días arbitrarios, mencionados algunos de ellos en la sección 5. Se realizaron un número pequeño de pruebas con documentos que sí se habían utilizado para el entrenamiento del modelo NER, con la intención de comprobar que detectaba correctamente textos con los que había sido entrenado.

# Pruebas unitarias y de integración

De las pruebas unitarias no se ha llevado un registro, ya que se han realizado continuamente: cada vez que **se modificaba una parte del desarrollo se comprobaba** el correcto funcionamiento de esta en la medida de lo posible. En las partes que afectaban a varios tipos de artículos, de campos o de cualquier entidad se probaba más de una vez.

Algo similar ocurría con las pruebas de integración: cada vez que se requerían más artículos para realizar más pruebas, para encontrar más observaciones de algunos campos, para probar nuevos desarrollos… estos **se obtenían realizando pruebas de integración hasta la fase que se quisiera probar**. De esta forma, para probar algo en la extracción, por ejemplo, se ejecutaba la ingesta de un día elegido de forma arbitraria, después la conversión a texto y a continuación la extracción de todo el día, por lo que se realizaba la extracción, tanto con el modelo NER como con las expresiones regulares de todos los artículos ingestados ese día.

Esta forma de trabajar aseguraba que no había ningún problema en desarrollos ya realizados derivado de los nuevos que se estaban probando, evitando posibles complicaciones más adelante.

Aunque no se ha guardado un registro concienzudo de estas pruebas, algunos de los días ingestados, convertidos y extraídos son los siguientes, de más antiguo a más reciente:

* **Anterior a 2020:** 11/04/2017, 14/03/2019, 12/12/2019, 24/12/2019.
* **Enero 2020:** 07/01/2020, 08/01/2020, 09/01/2020, 10/01/2020, 13/01/2020, 15/01/2020, 16/01/2020, 17/01/2020, 21/01/2020, 23/01/2020, 24/01/2020, 27/01/2020, 28/01/2020, 30/01/2020.
* **Febrero 2020:** 11/02/2020, 20/02/2020.
* **Marzo 2020:** 04/03/2020, 05/03/2020, 06/03/2020, 10/03/2020, 11/03/2020, 12/03/2020, 16/03/2020, 17/03/2020, 18/03/2020.
* **Abril 2020:** 10/04/2020.
* **Junio 2020:** 29/06/2020.
* **Julio 2020:** 15/07/2020.
* **Septiembre 2020:** 09/09/2020, 15/09/2020, 24/09/2020.
* **Octubre 2020:** 07/10/2020, 09/10/2020, 10/10/2020, 13/10/2020, 14/10/2020, 21/10/2020, 23/10/2020, 26/10/2020, 27/10/2020, 28/10/2020.
* **Noviembre 2020:** 11/11/2020, 12/11/2020, 13/11/2020, 16/11/2020, 17/11/2020, 18/11/2020, 19/11/2020, 20/11/2020, 27/11/2020.
* **Diciembre 2020:** 01/12/2020, 02/12/2020.

Se ha tratado de evitar realizar pruebas con documentos pertenecientes a otros corpus como el de análisis o el de etiquetado manual. Es por ello por lo que hay tantas pruebas con días recientes.

Además, desde el día 16 de diciembre hasta el día 21 de diciembre se realizó una **verificación** concienzuda tanto de identificación de ofertas (comprobar si todos los artículos que eran ofertas de empleo se detectaban) como de los campos detectados. Estas verificaciones fueron entregadas en formato Excel, con el estudio de la información de los campos y las observaciones de los valores incorrectos.

# Pruebas de aceptación

Se plantean una serie de **casos de prueba que abarcan el espectro de los desarrollos realizados**. La mayoría se va a centrar en la extracción de información a partir de los ficheros elegidos, ya que es el punto con más varianza y complejidad del pipeline. Para complementarlos, también habrá casos de prueba de ingesta, de conversión, de almacenamiento en la base de datos y de cierre de ofertas.

De los relativos a la extracción (incluida la acontecida en el cierre), se prueban 22 documentos, de los cuales 10 son artículos del BOA (45%), 8 son artículos del BOE (36%) y 4 son artículos de boletines provinciales aragoneses (18%).

En un entorno local, estas pruebas fueron ejecutadas por Oscar Potrony durante los días 2 y 3 de diciembre. En desarrollo, fueron ejecutadas por Oscar Potrony y por Alejandro Ciriano en el día 4 de diciembre. En preproducción, fueron ejecutadas en una reunión conjunta cuyos participantes fueron Oscar Potrony, Alejandro Ciriano, Sebastián Galbán, Julián Moyano y Gabriel Alcober.

(Nota: Puede que sea necesario activar el entorno virtual como se describe en el manual técnico).

## Caso de prueba de ingesta

### Caso 1

En este primer caso se va a probar el correcto funcionamiento de la ingesta de artículos de las secciones planteadas por defecto en los ficheros de configuración (2A y 2B tanto en BOA como en BOE). Se ha elegido un día en el que se obtienen tanto aperturas de ofertas como cierres de estas, y en el que se recupere boletines de todos los tipos abarcados.

**Parámetros de entrada:** Directorio base en el que se almacenarán los ficheros y día, en formato aaaammdd, del que se quieren recuperar los artículos.

**Datos de salida esperados:** Estructura de los artículos creada correctamente y toda la información guardada correctamente. A continuación, se muestra la estructura esperada, y se enumera la cantidad de ficheros esperada en cada directorio.

───20201116

│ BOA\_Sumarizado\_20201116\_1.xml

│ BOA\_Sumarizado\_20201116\_2.xml

│ BOE\_Sumario\_20201116.xml

│ BOPH\_Sumarizado\_20201116.xml

│ BOPT\_Sumarizado\_20201116.xml

│ BOPZ\_Sumarizado\_20201116.xml

│

├───apertura

│ ├───html

│ ├───info

│ ├───pdf

│ │ └───rotados

│ └───xml

│

└───cierre

│ ├───html

│ ├───info

│ ├───pdf

│ │ └───rotados

└───xml

En pdf, xml, html e info de apertura ha de haber 2 artículos del BOA, 28 del BOE, 1 del BOPH, 1 del BOPZ y 3 del BOPT. En rotados, dentro de pdf de apertura, ha de haber 28 artículos del BOE.

En pdf, xml, html e info de cierre ha de haber 2 artículos del BOA y 19 del BOE. En rotados de pdf de cierre, solo los 19 del BOE.

Todos estos ficheros con nombre con formato ‘fuente\_fecha\_número’.

**Pasos para realizar la prueba:**

(Primero se ha de pasar la carpeta de ingesta del backend al frontend, en el directorio */data/apps/ingesta-BO/pre/*.)

A continuación se detallan los pasos para crear un directorio en el que lanzar la ingesta y cómo lanzarla:

*cd /data/apps/ingesta-BO/pre/*

*mkdir Caso\_1*

*cd ..*

*./run.sh 20201116 /data/apps/ingesta-BO/pre/Caso\_1*

## Caso de prueba de conversión

### Caso 2

En este caso, y aprovechando todos los artículos obtenidos en el primero, consiste en convertir los mismos en ficheros con formato “.txt”, de los que poder obtener texto limpio en etapas posteriores.

**Parámetros de entrada:** Directorio base donde están almacenados los ficheros que se quieren convertir, con la estructura indicada en el Caso 1.

**Datos de salida esperados:** Creación de directorio ‘txt’ en ‘apertura’ y ‘cierre’, donde se encuentran los textos de todos los artículos ingestados con formato textual, con el mismo nombre (salvo el formato) que el fichero del que han obtenido la información.

**Pasos para realizar la prueba:**

*cd /data/apps/ingesta-BO/pre/conversion*

*python ./conversión\_a\_texto.py /data/apps/ingesta-BO/pre/casos\_prueba/Caso\_2/20201116*

## Casos de prueba de extracción y almacenamiento

Estos casos consisten en extraer la información deseada de distintos artículos de los boletines tratados. Con ese objetivo, el programa utiliza, de cada artículo, su fichero textual y su fichero de información. Este último es modificado en el proceso, con la incorporación de nueva información. Por ello, se proporciona una copia del mismo en caso de querer volver a lanzar la prueba (habría que eliminar el fichero de información, copiar y pegar la copia, y renombrarla a como estaba el fichero anterior).

En los diferentes casos se van a mencionar las condiciones a las que va enfocada cada uno, para comprobar todo lo abarcado.

**Los pasos para realizar todos los casos** de esta sección son dirigirse al directorio donde se encuentra el fichero de extracción e invocar el programa con el caso correspondiente:

*cd /data/apps/ingesta-BO/pre/extraccion*

*python ./extraccion.py X*

(siendo X el número del caso que se quiera probar, entre 3 y 22).

Además, en el caso de querer almacenar el resultado para comprobar la salida en la propia base de datos, los pasos a seguir, una vez ha terminado el proceso de extracción, son:

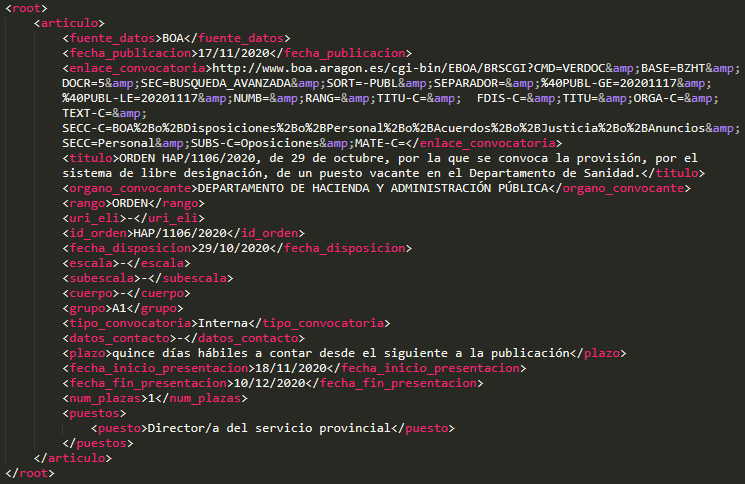
cd /data/apps/ingesta-BO/pre/almacenamiento

python ./almacenamiento.py X

**Los pasos para comprobar la inserción en la base de datos de todos estos casos** se van a indicar en cada uno de los casos, con el apoyo de las vistas creadas.

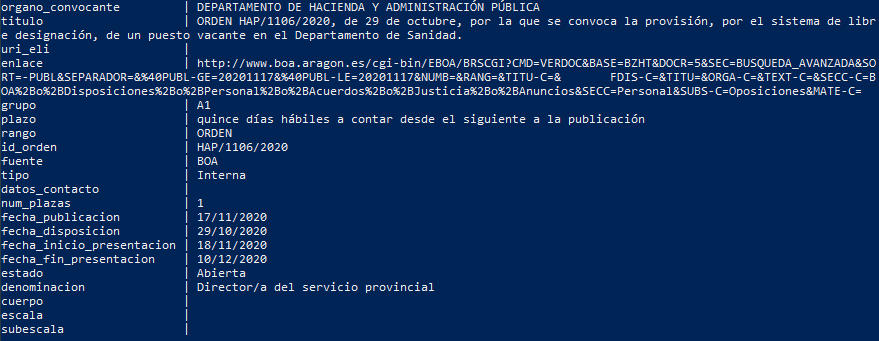
### Caso 3

Este primer caso está enfocado en la detección de puesto ofertado en artículos del BOA donde éste venga precedido por la etiqueta “Denominación:”. Además, pueden verse detectados otros campos como grupo, tipo de convocatoria, el plazo o las fechas. El fichero de información quedaría:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'DEPARTAMENTO DE HACIENDA Y ADMINISTRACIÓN PÚBLICA'  AND FECHA\_DISPOSICION = '29/10/2020'; |



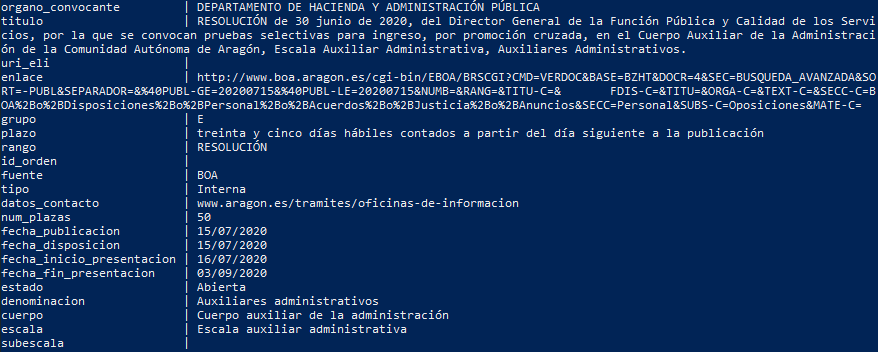
### Caso 4

En este caso, se obtienen cuerpo, escala y puesto del artículo del BOA a partir de su título. Además, en este caso extrae correctamente el número de plazas totales e incluso el plazo y sus fechas, que en este caso está conformado por un número de días determinado por más de una palabra. Es decir, ha detectado correctamente “treinta y cinco” y no únicamente la última palabra, “cinco”. El fichero final quedaría:



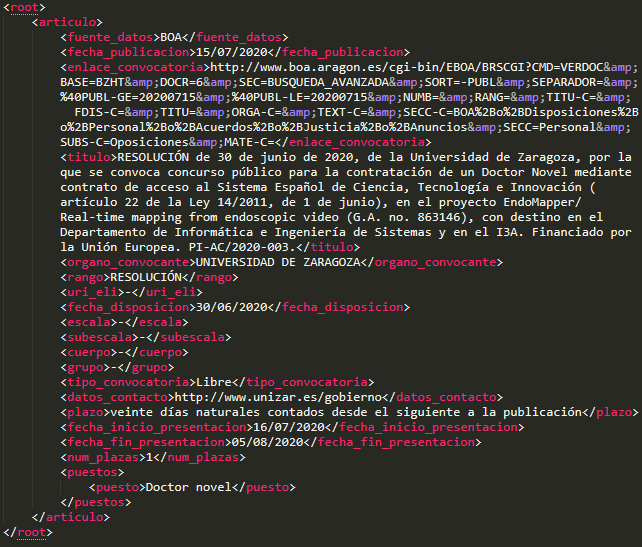
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'DEPARTAMENTO DE HACIENDA Y ADMINISTRACIÓN PÚBLICA'  AND FECHA\_DISPOSICION = '15/07/2020'; |



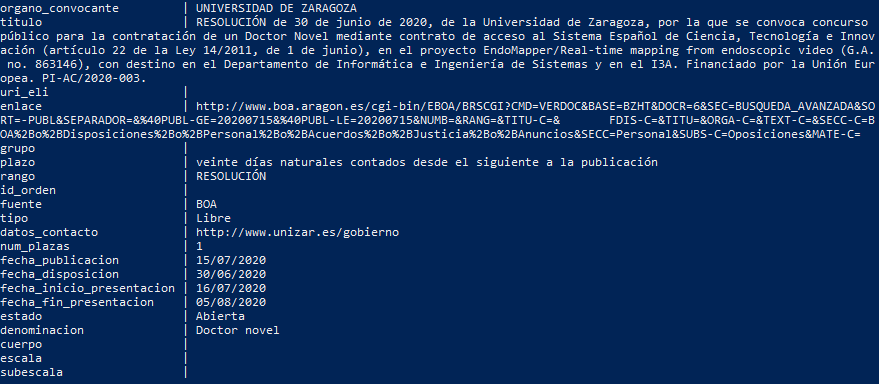
### Caso 5

A diferencia de los dos casos anteriores, en este artículo del BOA se va a obtener el puesto ofertado sin venir en una ubicación fija que se haya contemplado en las expresiones regulares. También es el primer caso cuyo tipo es libre y cuyo plazo indica días naturales. El fichero de configuración quedaría como sigue:



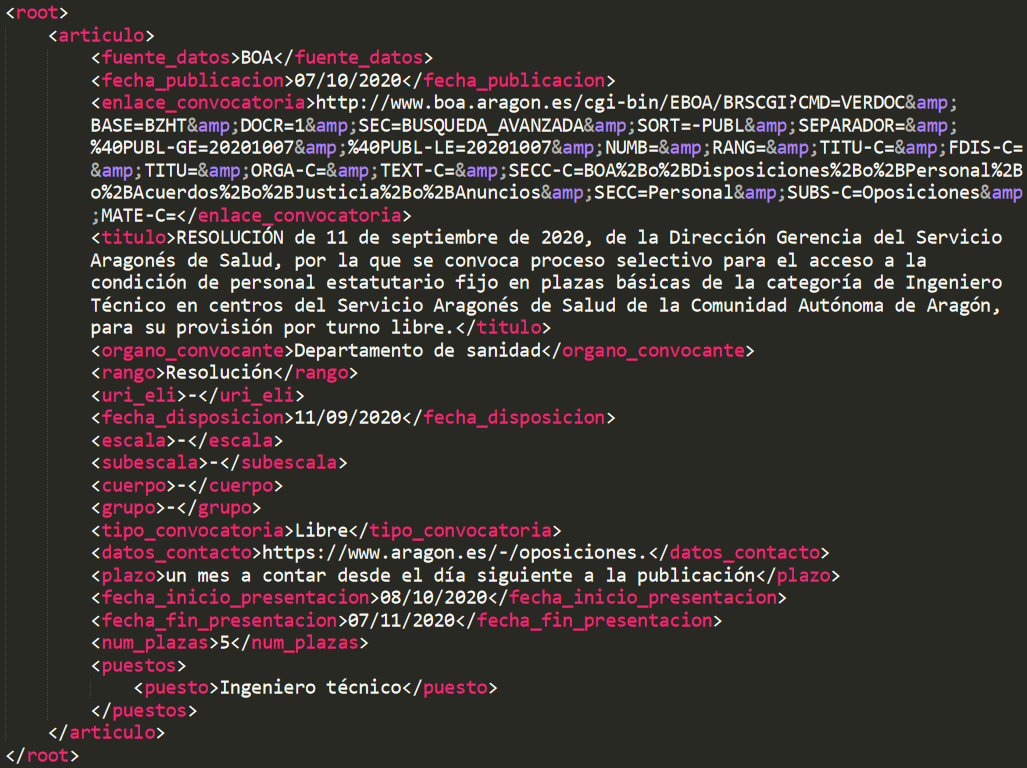
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'UNIVERSIDAD DE ZARAGOZA'  AND FECHA\_DISPOSICION = '30/06/2020'; |



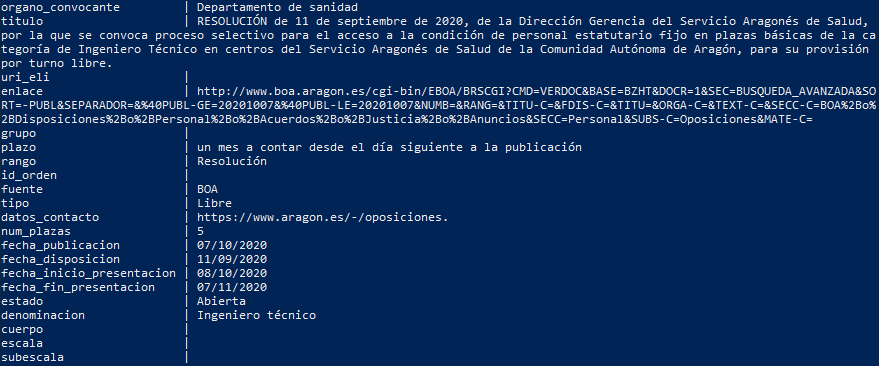
### Caso 6

En este caso, el número de días del plazo de presentación de solicitudes viene dado en formato de mes, y no de días como es más habitual. La salida esperada es:



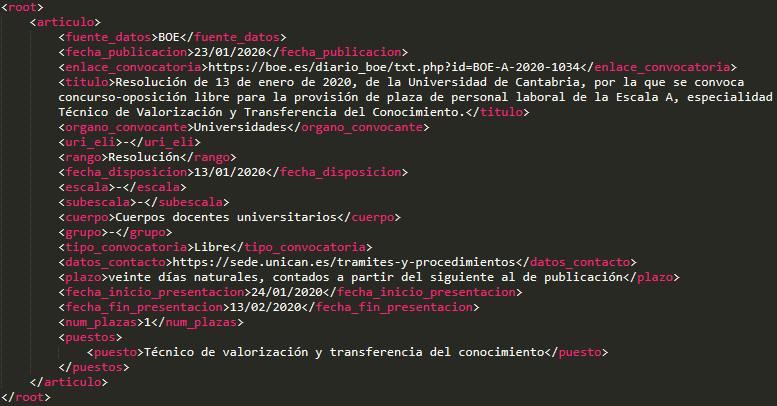
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'Departamento de sanidad'  AND FECHA\_DISPOSICION = '11/09/2020'; |



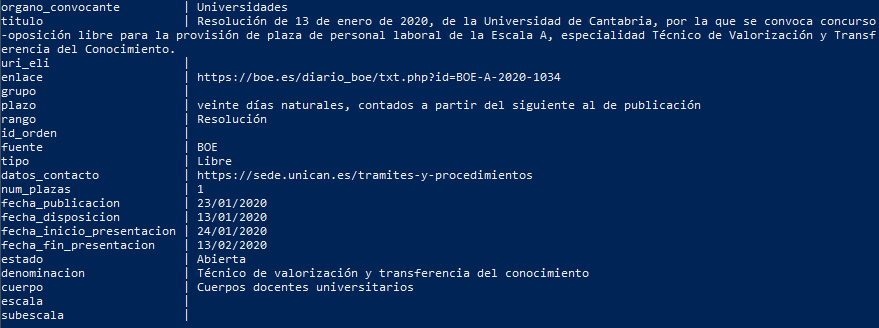
### Caso 7

Pasando a probar algún documento del BOE, en este primer caso se detecta el puesto gracias a venir fijado por la palabra “Especialidad” en el título. Existen ciertas cadenas que se han observado que sirven de esta forma, como “Identificación de la plaza”. La salida esperada es:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Universidades'  AND FECHA\_DISPOSICION = '13/01/2020'; |



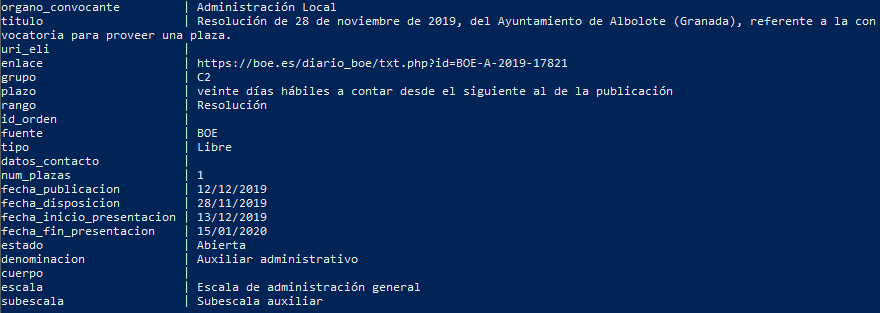
### Caso 8

En este caso, se va a obtener un puesto del BOE que viene suelto en el texto, sin tener palabras fijas alrededor que permita obtenerlo mediante expresiones regulares. Además, es un artículo del 2019. También detecta otros atributos como escala, subescala o grupo. La salida esperada del fichero de información es:



Para comprobar la inserción en la base de datos, se ejecuta la siguiente consulta, que requiere de una vista diferente a la histórica de 2020:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_ABIERTAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Administración Local'  AND FECHA\_DISPOSICION = '28/11/2019'; |



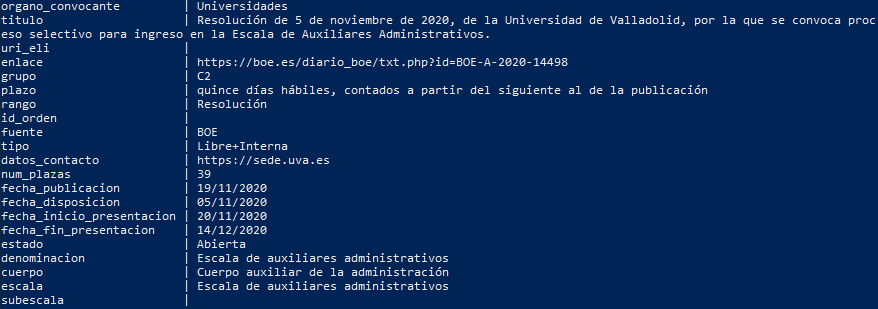
### Caso 9

En este caso, también del BOE, no se detecta ningún puesto ofertado. Sin embargo, se detecta la escala, que es en lo que realmente se centra la convocatoria (y por ello se va a indicar como puesto). También detecta bien el número total de plazas (apareciendo con un texto compuesto en el documento y apareciendo también cifras parciales). Por otra parte, detecta el tipo de convocatoria “Libre+Interna”. La salida queda como lo que sigue:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Universidades'  AND FECHA\_DISPOSICION = '05/11/2020'; |



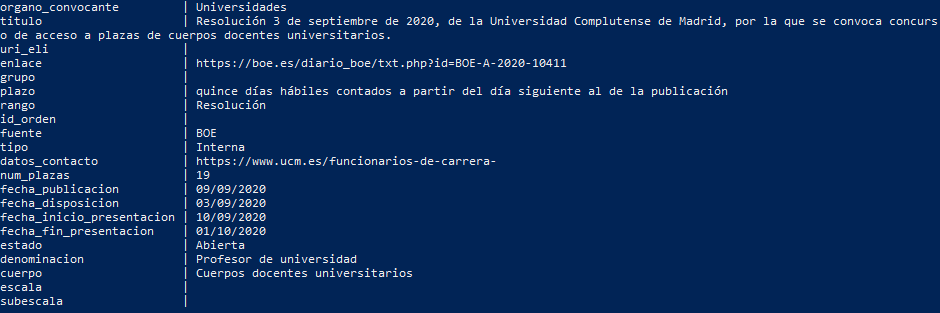
### Caso 10

Debido a la posible aparición de puestos que no son los ofertados, excepto en los artículos que cuentan con tablas de puestos se coge un único puesto (el encontrado en la parte más superior del artículo). Por esta razón, en casos muy específicos como el de esta prueba se obtiene un único puesto, aunque aparezcan más enumerados. La salida esperada del fichero de información es la siguiente:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Universidades'  AND FECHA\_DISPOSICION = '03/09/2020'; |



### Caso 11

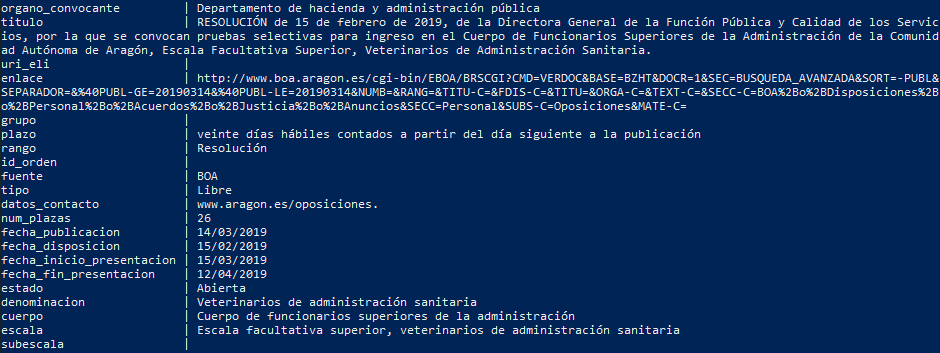
En este caso se va a probar un artículo del BOA publicado en marzo de 2019, con la intención de probar si artículos de hace un año son similares y, por tanto, recuperan bien la información.

La salida esperada del fichero de configuración tras ejecutar la extracción es:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_ABIERTAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'Departamento de hacienda y administración pública'  AND FECHA\_DISPOSICION = '15/02/2019'; |



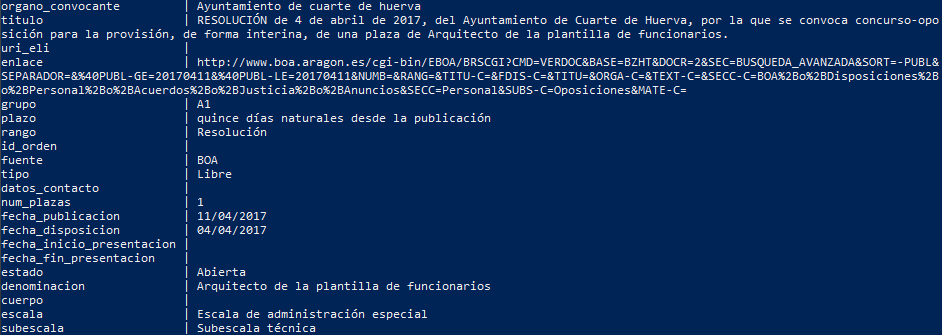
### Caso 12

También se quiere comprobar que, sobre todo para los artículos del BOA, trabaja bien para años más anteriores. Por ello, en este caso se utiliza un artículo del 2017. La salida esperada es:



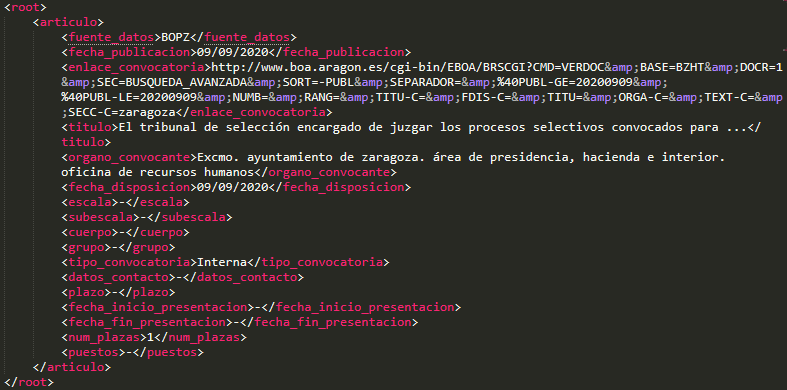
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_ABIERTAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'Ayuntamiento de cuarte de huerva'  AND FECHA\_DISPOSICION = '04/04/2017'; |



### Caso 13

Este caso va a tratar sobre un cambio en el umbral de confianza para insertar boletines. Cuando se realiza el almacenamiento de las ofertas de un artículo, se comprueba el número de campos obtenidos para estas. Si no llegan al umbral indicado en el fichero de configuración ‘auxiliar.xml’, las ofertas del artículo no son insertadas. Para guardar los artículos no insertados, se escribe además su nombre en el fichero ‘articulos\_no\_insertados.log’, ubicado en el mismo directorio en el que se lanza el programa. En este caso, se va a extraer la información de un artículo (que se ha identificado como oferta por su contenido, aunque realmente no es una oferta), como en los demás casos de prueba. La salida esperada es la siguiente:

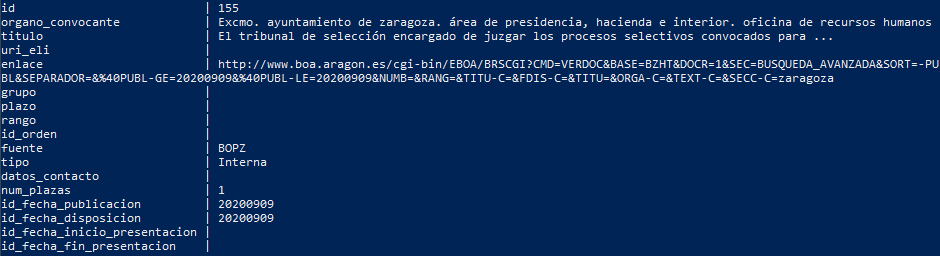


Como se puede observar, únicamente se han detectado 8 casos, número inferior al indicado actualmente en el campo ‘num\_min\_campos’ en el fichero de configuración ‘auxiliar.xml’. Por ello, ejecutando la secuencia de almacenamiento no se almacena nada, como se puede comprobar con la consulta siguiente (devuelve *0 rows*):

|  |
| --- |
| SELECT \*  FROM CONVOCATORIA  WHERE ORGANO\_CONVOCANTE = 'Excmo. ayuntamiento de zaragoza. área de presidencia, hacienda e interior. oficina de recursos humanos'; |

Además, se habrá creado un fichero llamado articulos\_no\_insertados.log en nuestro directorio actual, cuya primera línea indicará ‘BOPZ\_20200909\_1’, por lo que se guarda un registro de estos ficheros, en caso de querer consultarlos.

Sin embargo, si a continuación se cambia, del fichero ‘auxiliar.xml’, ubicado en el directorio ‘ficheros\_configuracion’, el valor de ‘num\_min\_campos’ por 8 y volvemos a ejecutar la secuencia de almacenamiento, podremos ver las inserciones con la misma consulta descrita.



Nota: Recordar volver a dejar ‘num\_min\_campos’ a su valor original, 11.

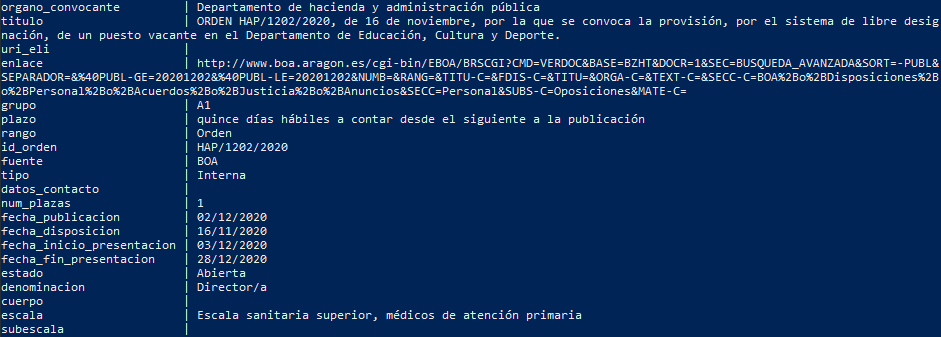
### Caso 14

También hay artículos de apertura de ofertas de empleo que tienen rango de orden, en lugar de ser resoluciones. Esta prueba trata sobre un artículo de BOA de rango orden, que detecte a su vez la propia ID de la misma. La salida esperada es:



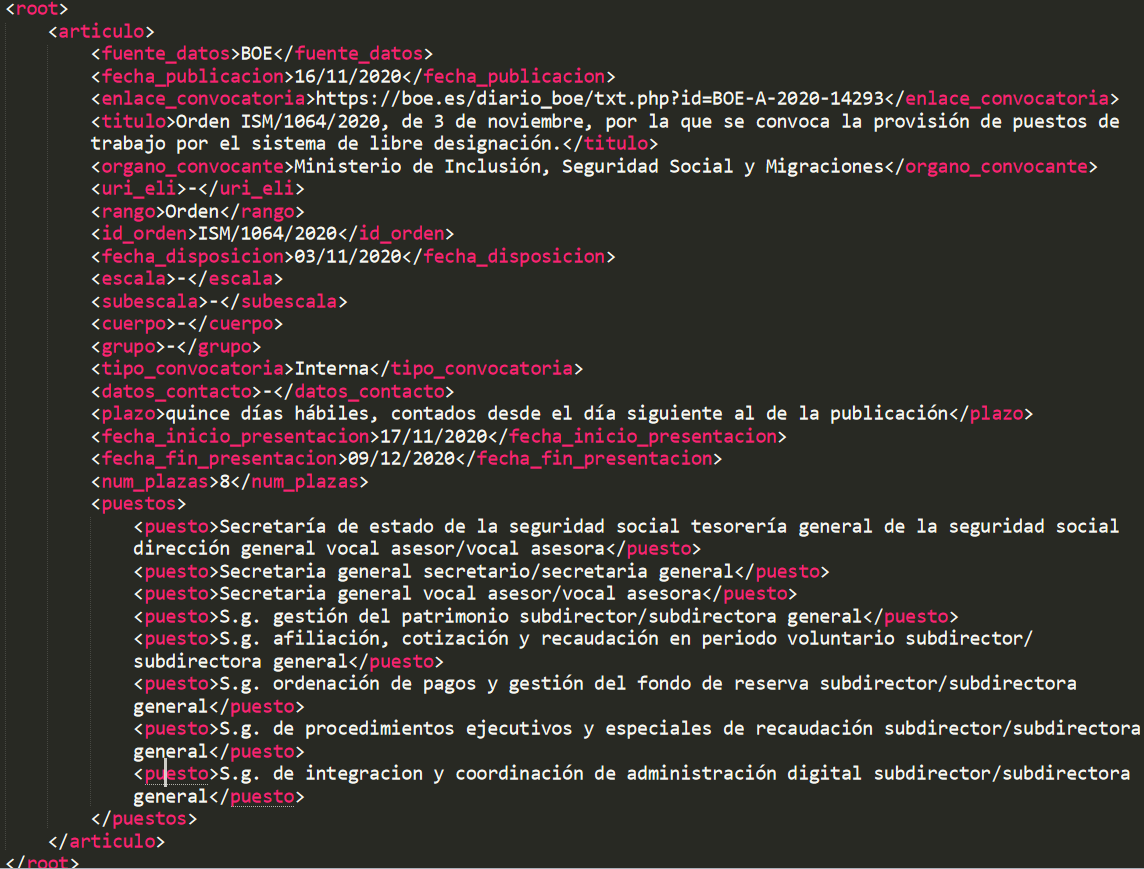
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE ID\_ORDEN = 'HAP/1202/2020'; |



### Caso 15

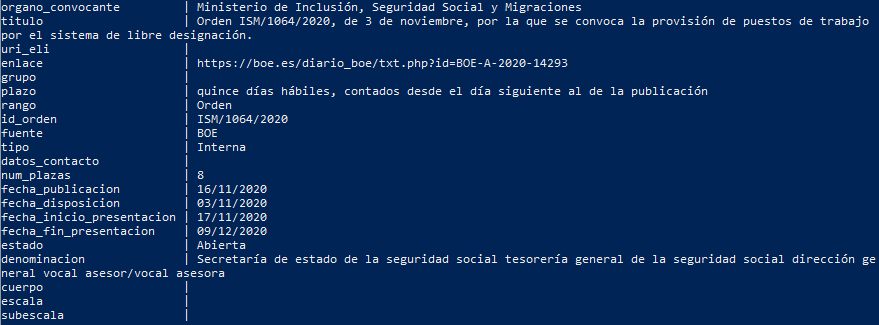
En este caso se van a obtener múltiples puestos, obtenidos de las tablas de un anexo de un artículo del BOE, donde estas tablas vienen con formato horizontal. Como se puede observar coge toda la celda como puesto, y en algunos casos como el presente, se ha formateado la tabla metiendo información extra en la propia celda. El fichero de información quedaría como sigue:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

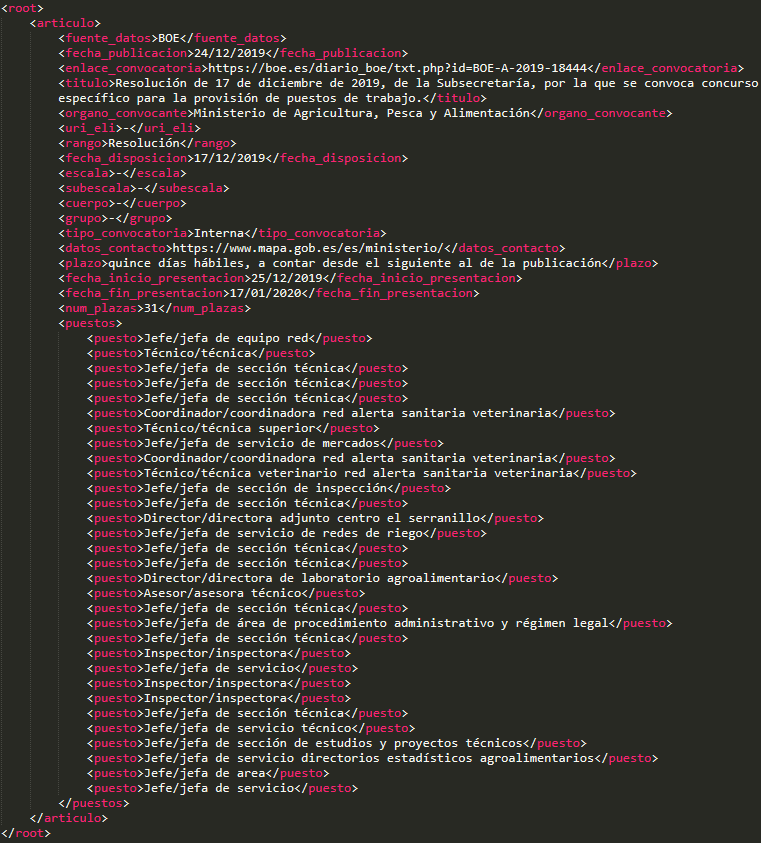
|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Ministerio de Inclusión, Seguridad Social y Migraciones'  AND FECHA\_DISPOSICION = '03/11/2020'; |

La salida obtenida en este caso serían 8 registros como el que aparece en la siguiente imagen, donde solo cambiaría de un registro a otro la denominación del puesto. Los 8 registros contendrían cada uno de los puestos aparecidos en la imagen del fichero de información que se ha mostrado.



### Caso 16

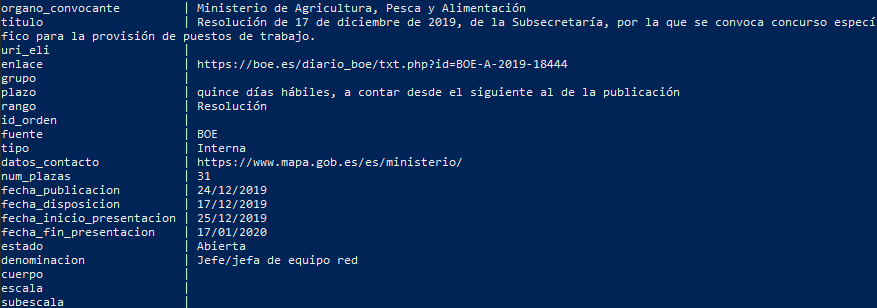
Este caso, similar al anterior, se diferencia con este en que las tablas vienen inicialmente con un formato vertical (a -90º). Detecta correctamente los 31 puestos que aparecen. Este fichero, además, es de 2019 (por lo que habrá que elegir otra vista para comprobar su inserción). El fichero de información queda:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

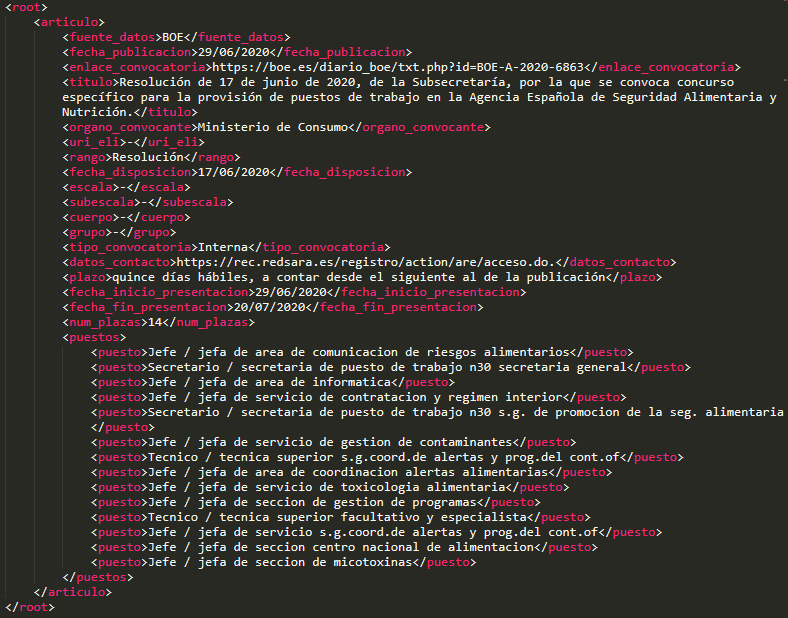
|  |
| --- |
| SELECT \*  FROM OFERTAS\_ABIERTAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Ministerio de Agricultura, Pesca y Alimentación'  AND FECHA\_DISPOSICION = '17/12/2019'; |

Al igual que en el Caso 15, aparecen múltiples registros (en este caso 31) con la misma información salvo la denominación del puesto, que varía según los puestos mostrados en la imagen anterior.



### Caso 17

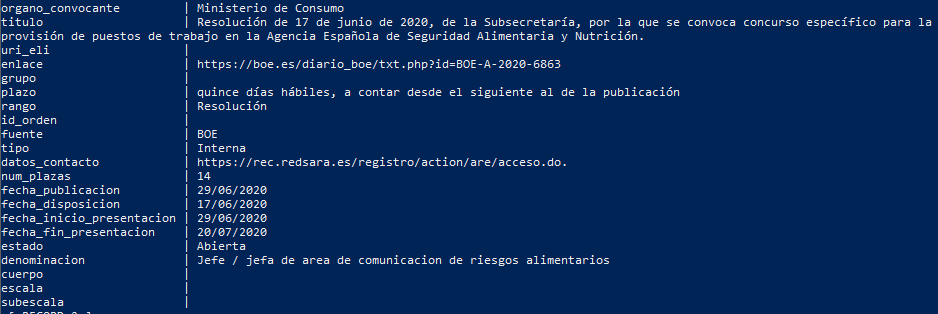
Esta prueba, similar al Caso 16, obtiene puestos de tablas que inicialmente vienen con formato vertical, a -90º. Sin embargo, en este caso no se obtienen todos los puestos aparecidos en estas. Esto ocurre debido a que las tablas no son cerradas por debajo con líneas, de forma que él puesto de la última celda de cada tabla no es recuperado, ya que la herramienta no trata esa fila como parte de la tabla. Es un problema de la herramienta utilizada para hacer esas tablas en concreto, por parte del creador del artículo.



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

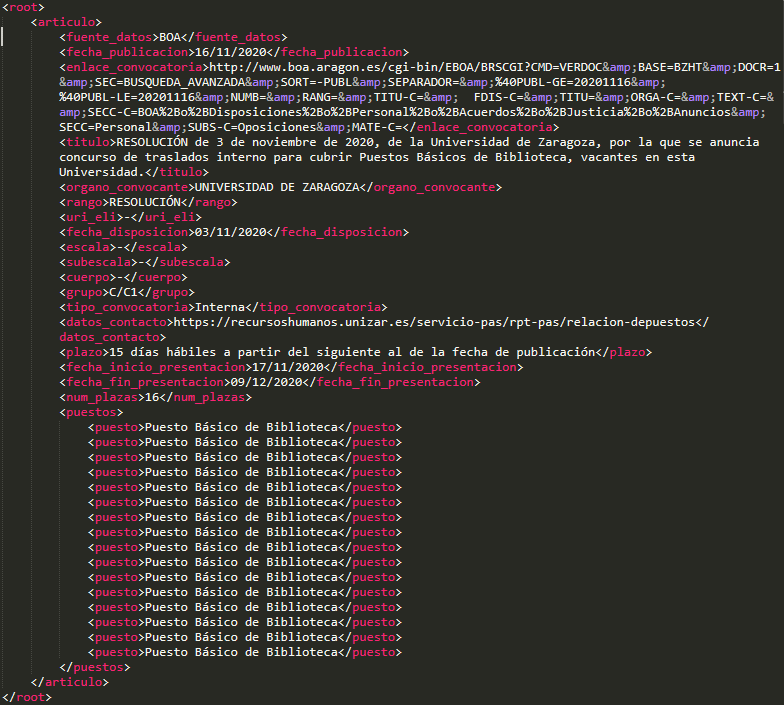
|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Ministerio de Consumo'  AND FECHA\_DISPOSICION = '17/06/2020'; |

La salida obtenida en este caso serían 14 registros como el que aparece en la siguiente imagen, donde solo cambiaría de un registro a otro la denominación del puesto. Los 14 registros contendrían cada uno de los puestos aparecidos en la imagen del fichero de información que se ha mostrado.



### Caso 18

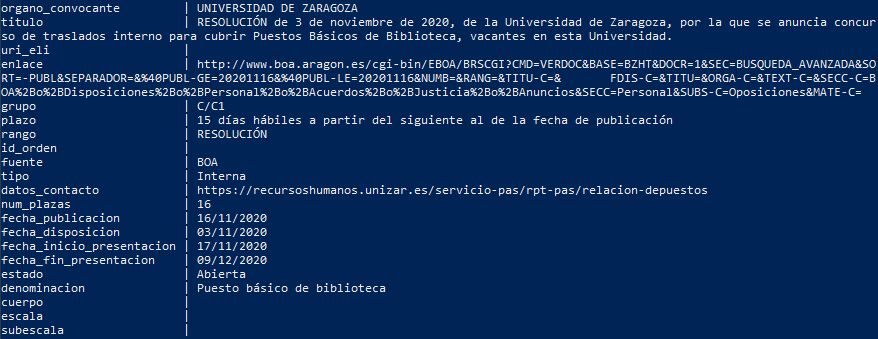
Este caso tiene como particularidad que ha de obtener los puestos de una tabla en el anexo en un artículo del BOA. Es un caso visto en muy pocas ocasiones. El fichero de información quedaría:



Para comprobar la inserción en la base de datos, se ejecuta la consulta:

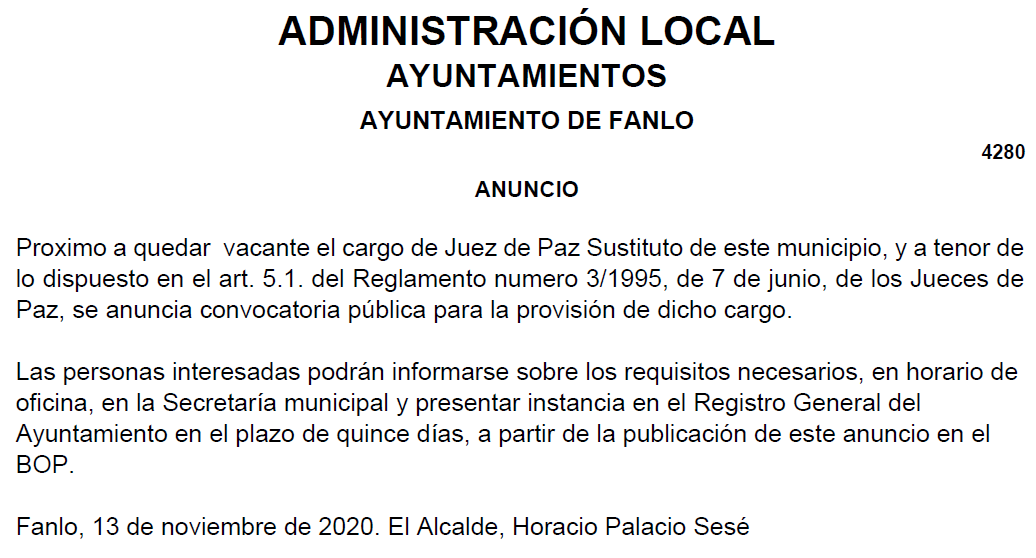
|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'UNIVERSIDAD DE ZARAGOZA'  AND FECHA\_DISPOSICION = '03/11/2020'; |

La salida de este caso serán 16 ofertas iguales inicialmente, siendo cada una como la que sigue:

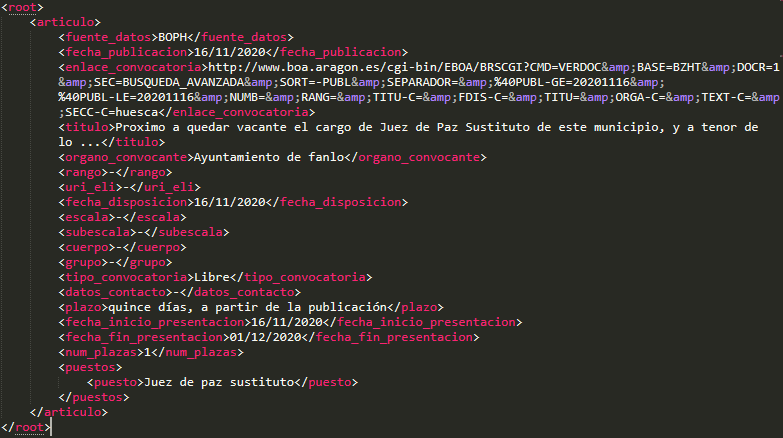


### Caso 19

Pasando a los boletines provinciales, para este caso se va a utilizar un boletín provincial de Huesca. El caso que acontece tiene de especial lo escueto que es (8 líneas), y sin embargo encuentra correctamente 12 campos (todos los que aparecen), por lo que incluso supera el umbral definido por defecto para su inserción en la base de datos. Por lo corto que es se adjunta el documento completo a continuación:

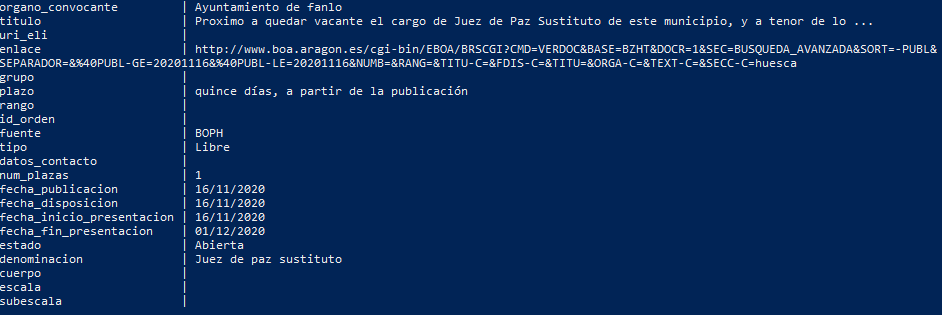


Tras la extracción, la salida esperada es la siguiente:



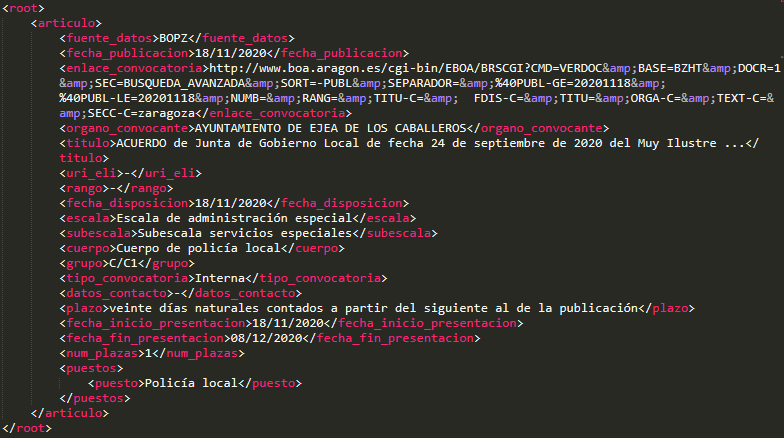
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOPH'  AND ORGANO\_CONVOCANTE = 'Ayuntamiento de fanlo'  AND FECHA\_DISPOSICION = '16/11/2020'; |



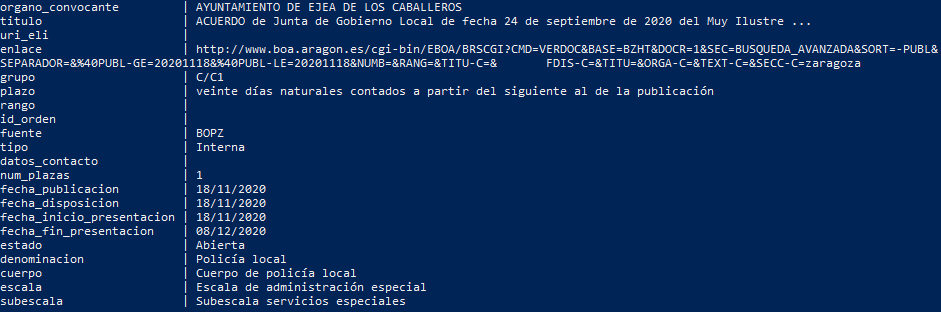
### Caso 20

En este caso, se va a tratar un boletín provincial largo, con estructura más similar a artículos del BOA o del BOE. Va a ser un artículo del boletín de la provincia de Zaragoza. La salida esperada en el fichero de información es la siguiente, que incluye también una subescala:



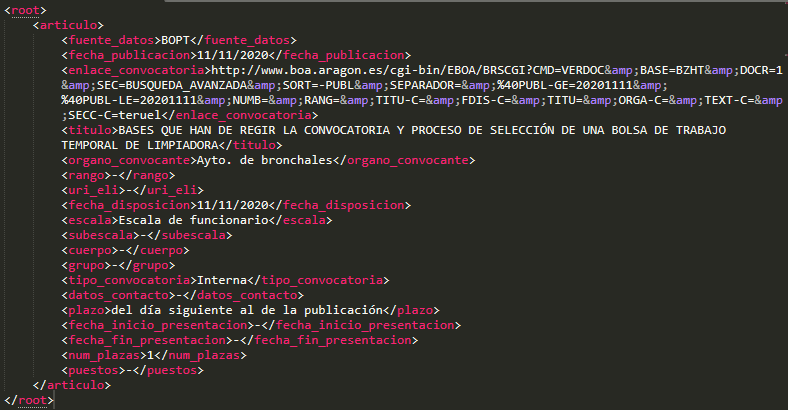
Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOPZ'  AND ORGANO\_CONVOCANTE = 'AYUNTAMIENTO DE EJEA DE LOS CABALLEROS'  AND FECHA\_DISPOSICION = '18/11/2020'; |



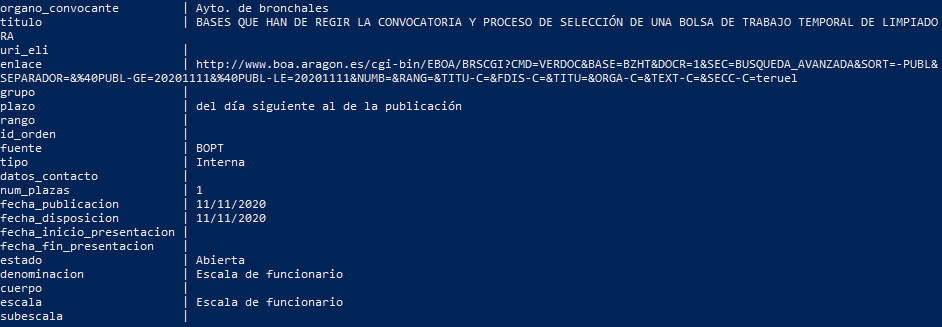
### Caso 21

Sin embargo, no siempre se recogen bien los campos de los boletines. Un ejemplo sería este caso, que trata sobre un artículo del boletín provincial de Teruel. Dado que en los boletines provinciales se piden puestos más específicos que en BOA o BOE, es más probable que el modelo no los conozca, por lo que no sea capaz de discernir que lo son. Además, debido a lo peor trabajado que están estos artículos, su estructura puede complicar más la extracción de otros campos, como el plazo. Por otra parte, otros campos sí han sido detectados correctamente, como el número de plazas. La salida esperada para este artículo es la siguiente:



Debido al umbral que se ha determinado, este artículo pasa la criba, por lo que se inserta en la base de datos. Para comprobar esto, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOPT'  AND ORGANO\_CONVOCANTE = 'Ayto. de bronchales'  AND FECHA\_DISPOSICION = '11/11/2020'; |



## Caso de prueba de cambios en los ficheros de configuración

### Caso 22

En este caso, se va a realizar la ingesta tanto con las etiquetas por defecto como eliminando una etiqueta para los artículos del BOA y añadiendo otra para los mismos.

Para ello, primero se ejecuta:

A continuación se detallan los pasos para crear un directorio en el que lanzar la ingesta y cómo lanzarla:

*cd /data/apps/ingesta-BO/pre/*

*mkdir Caso\_22*

*cd Caso\_22*

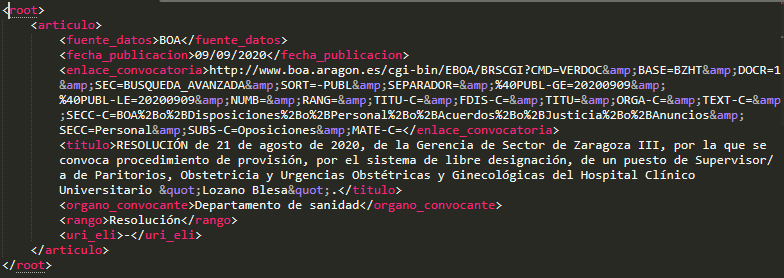
*mkdir Sin\_Cambio*

*mkdir Con\_Cambio*

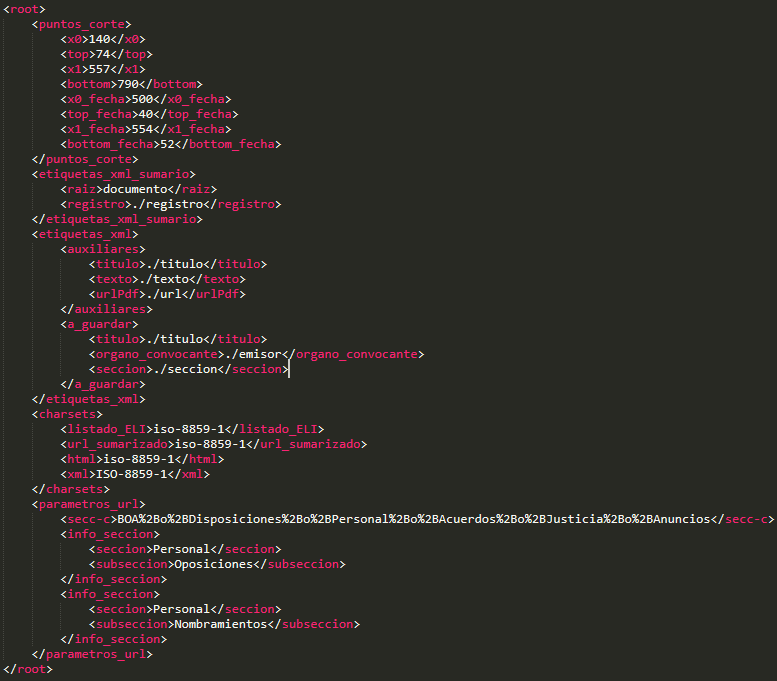
*cd ../..*

*./run.sh 20200909 /data/apps/ingesta-BO/pre/Caso\_22/Sin\_Cambio*

De esta forma, se creará una estructura como la del Caso 1, y se guardará una serie de artículos. Poniendo la atención en el del BOA de apertura, la salida esperada de su fichero de información es:



Ahora, en el fichero de configuración ‘BOA\_conf.xml’, ubicado en el directorio ‘ficheros\_configuracion’, se va a eliminar la etiqueta <rango> de etiquetas\_xml/a\_guardar, y se va a añadir en esa misma ubicación la etiqueta <seccion>, con el valor ‘./seccion’, quedando finalmente como en la imagen:

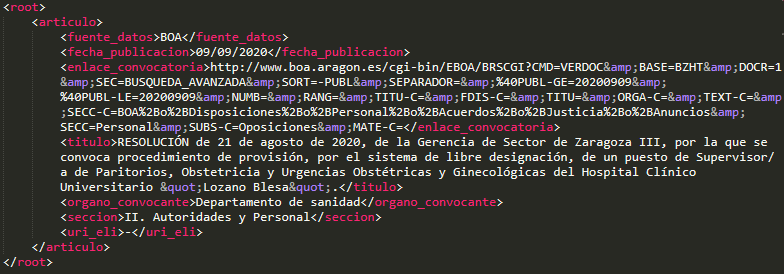


A continuación, se ejecuta la ingesta otra vez, ahora apuntando a la carpeta Con\_Cambio, para poder ver las diferencias:

*cd /data/apps/ingesta-BO/*

*./run.sh 20200909 /data/apps/ingesta-BO/pre/Caso\_22/Con\_Cambio*

Ahora, como podemos observar, en el nuevo fichero de información la salida es diferente, ya que no aparece el rango pero sí la sección:



Cabe señalar que es posible eliminar algunas etiquetas como el rango pero no otras que tienen mayor importancia en las operaciones siguientes, como el título o el órgano convocante. Las etiquetas que se pueden incorporar vienen dadas por la información que aparece en los xml de cada tipo de boletín, pudiendo consultarse en cualquiera de los mismos. Además, modificar las etiquetas a guardar en un tipo de boletín no repercute en los demás, ya que se han independizado en los ficheros de configuración. Por otra parte, recordar que la sección no se incorporará a la base de datos, ya que no es un atributo esperado por la misma. Sin embargo, los demás atributos (salvo rango), se incorporan correctamente si se sigue el flujo (pasando la información al backend).

Nota: Se recomienda devolver al fichero ‘BOA\_conf.xml’ a su estado original.

## Casos de pruebas cierre de ofertas

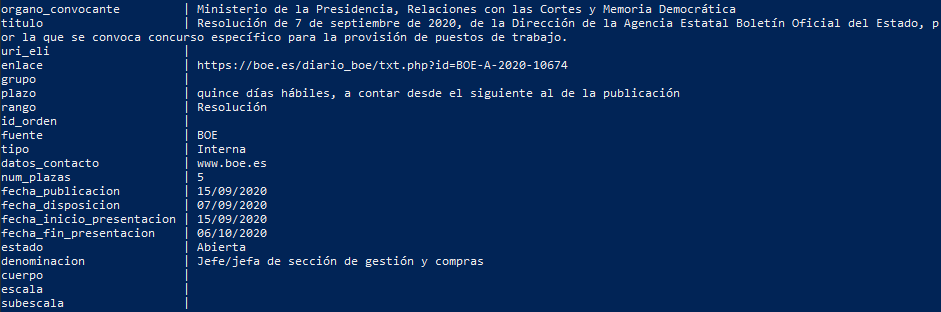
### Caso 23

Este caso es un cierre exitoso de cierres de ofertas del BOE. Primero, se ejecuta la extracción y el almacenamiento de Caso\_23\_apertura (siendo el número del caso utilizado en la extracción ‘23\_apertura’) que, además, obtiene los puestos de una tabla vertical. La salida deseada de su fichero de info es:

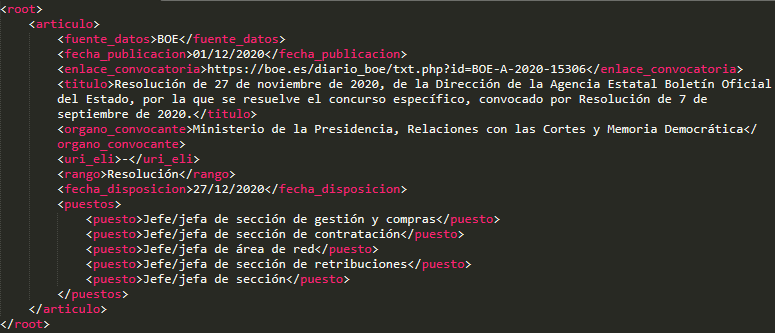


Se almacenan 5 ofertas, cada una referenciando a un puesto diferente, pero todas referenciando a la misma convocatoria. La salida esperada de la siguiente consulta son 5 registros equivalentes, donde solo varía la denominación. La consulta es:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Ministerio de la Presidencia, Relaciones con las Cortes y Memoria Democrática'  AND FECHA\_DISPOSICION = '07/09/2020'; |



Una vez extraídas y almacenadas las ofertas, se va a proceder a extraer y utilizar la información de Caso\_23\_cierre, que contiene el artículo del BOE que cierra todas las ofertas abiertas por el anterior, esta vez indicando los puestos en una tabla horizontal. Para ello, primero se extrae la información como indicado arriba, siendo el número del caso ‘23\_cierre’. La salida esperada para su fichero de información es la siguiente:



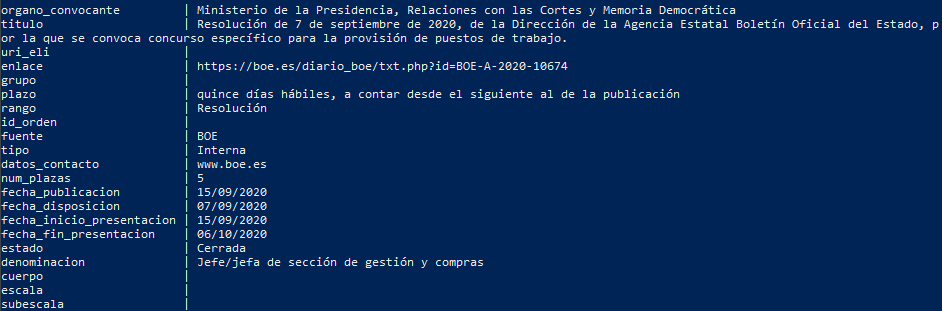
A continuación, se lanza el script de cierre de ofertas para cerrar las coincidentes:

*cd /data/apps/ingesta-BO/pre/cierres*

*python ./cierres\_convocatorias.py 23\_cierre*

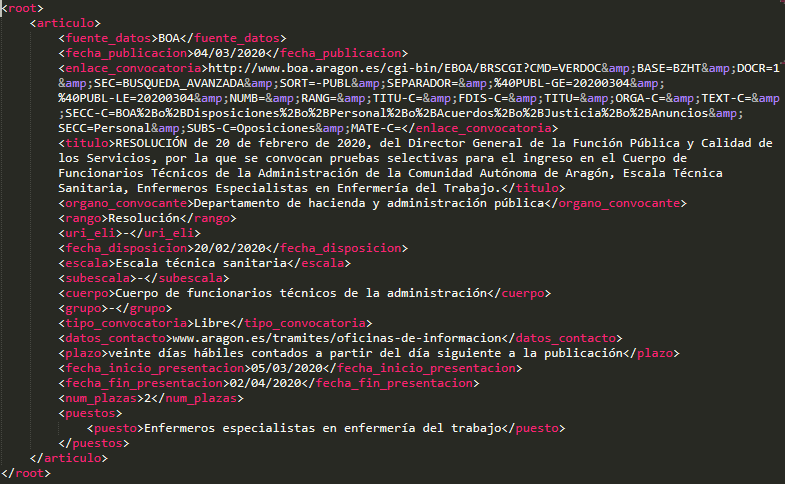
Una vez ejecutado, se lanza la siguiente consulta y, al igual que en el caso anterior, saldrán 5 resultados iguales excepto por la denominación. Se puede ver que su estado ha pasado a ‘Cerrada’.

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOE'  AND ORGANO\_CONVOCANTE = 'Ministerio de la Presidencia, Relaciones con las Cortes y Memoria Democrática'  AND FECHA\_DISPOSICION = '07/09/2020'; |



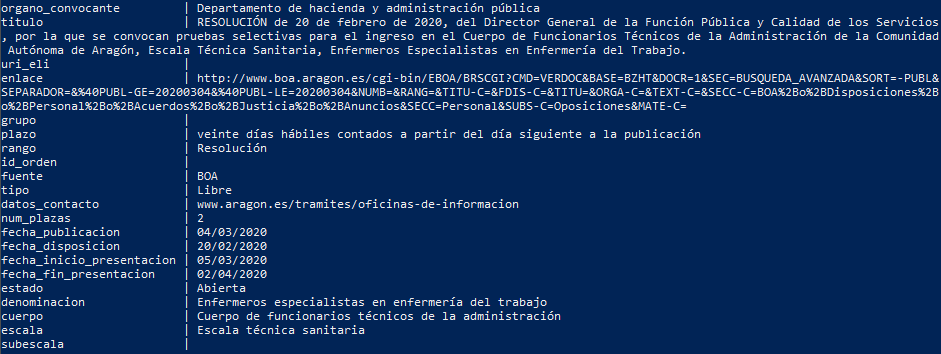
### Caso 24

Esta prueba es otro caso exitoso de cierre de ofertas, esta vez en el BOA. Además, este caso trata sobre una convocatoria cerrada porque se ha dejado sin efecto, a diferencia de la anterior que la cerraba porque se habían adjudicado los puestos. Al ejecutar la extracción de ‘24\_apertura’, la salida esperada de su fichero xml es:

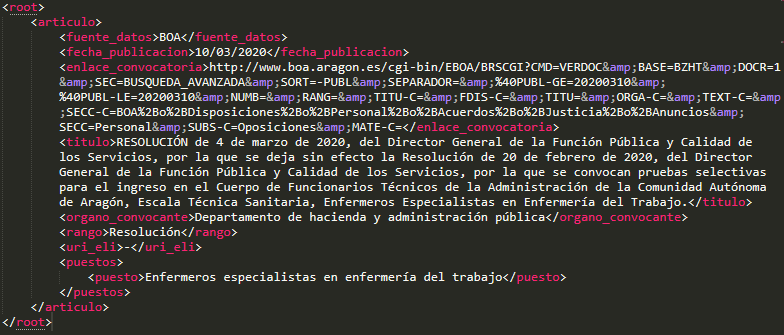


Tras almacenarlo en la base de datos, se puede comprobar con la siguiente consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_ABIERTAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'Departamento de hacienda y administración pública'  AND FECHA\_DISPOSICION = '20/02/2020'; |

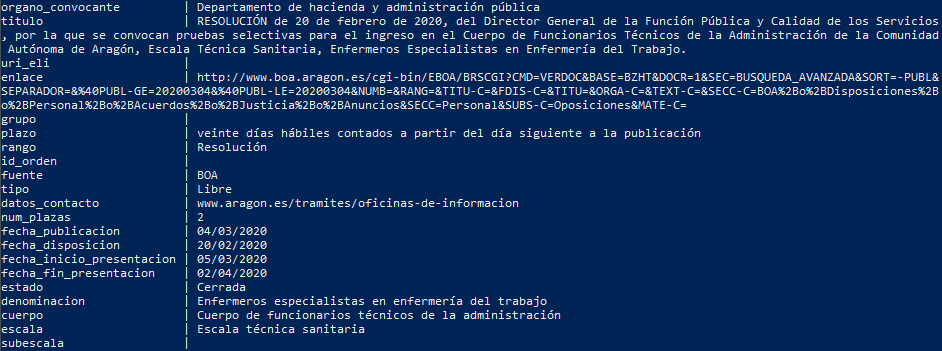


A continuación, se procede a cerrarla. Para ello, primero se realiza la extracción del caso “24\_cierre”, cuya salida esperada es:



A continuación, se lanza el script de cierre de ofertas para cerrar las coincidentes de la misma forma que en el Caso 23, cambiando ‘23\_cierre’ por ‘24\_cierre’. Esto habría cerrado la oferta en la base de datos, por lo que se procede a comprobarlo:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE FUENTE = 'BOA'  AND ORGANO\_CONVOCANTE = 'Departamento de hacienda y administración pública'  AND FECHA\_DISPOSICION = '20/02/2020'; |



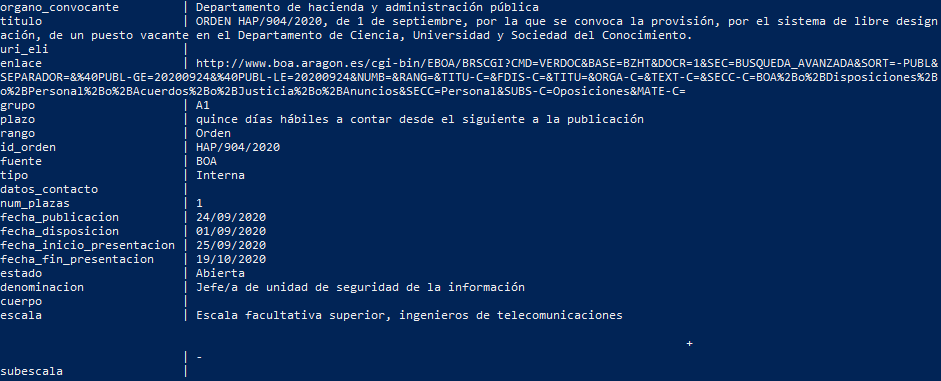
### Caso 25

Sin embargo, los cierres de ofertas no siempre son exitosos. Por ello, esta prueba no va a cerrar correctamente la oferta aparecida en artículos del BOA. Se trata de un puesto que no viene redactado de la misma manera, por lo que el modelo no puede dar por hecho que se trate del mismo. Ejecutando la extracción del caso ‘25\_apertura’ la salida esperada es la siguiente:

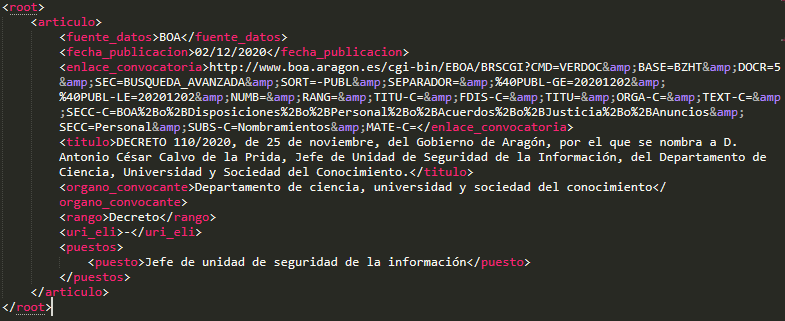


Para comprobar la inserción en la base de datos, se ejecuta la consulta:

|  |
| --- |
| SELECT \*  FROM OFERTAS\_ABIERTAS  WHERE ID\_ORDEN = 'HAP/904/2020'; |



Después se procede a cerrarla. Para ello, primero se realiza la extracción del caso “24\_cierre”, cuya salida esperada es:



A continuación, se lanza el script de cierre de ofertas para cerrar las coincidentes de la misma forma que en el Caso 23, cambiando ‘23\_cierre’ por ‘25\_cierre’. Al lanzar la siguiente consulta nos sigue apareciendo el estado de la oferta ‘Abierta’, debido a ese cambio entre la apertura y el cierre de ‘Jefe/a’ a ‘Jefe’.

|  |
| --- |
| SELECT \*  FROM OFERTAS\_HISTORICAS  WHERE ID\_ORDEN = 'HAP/904/2020'; |

