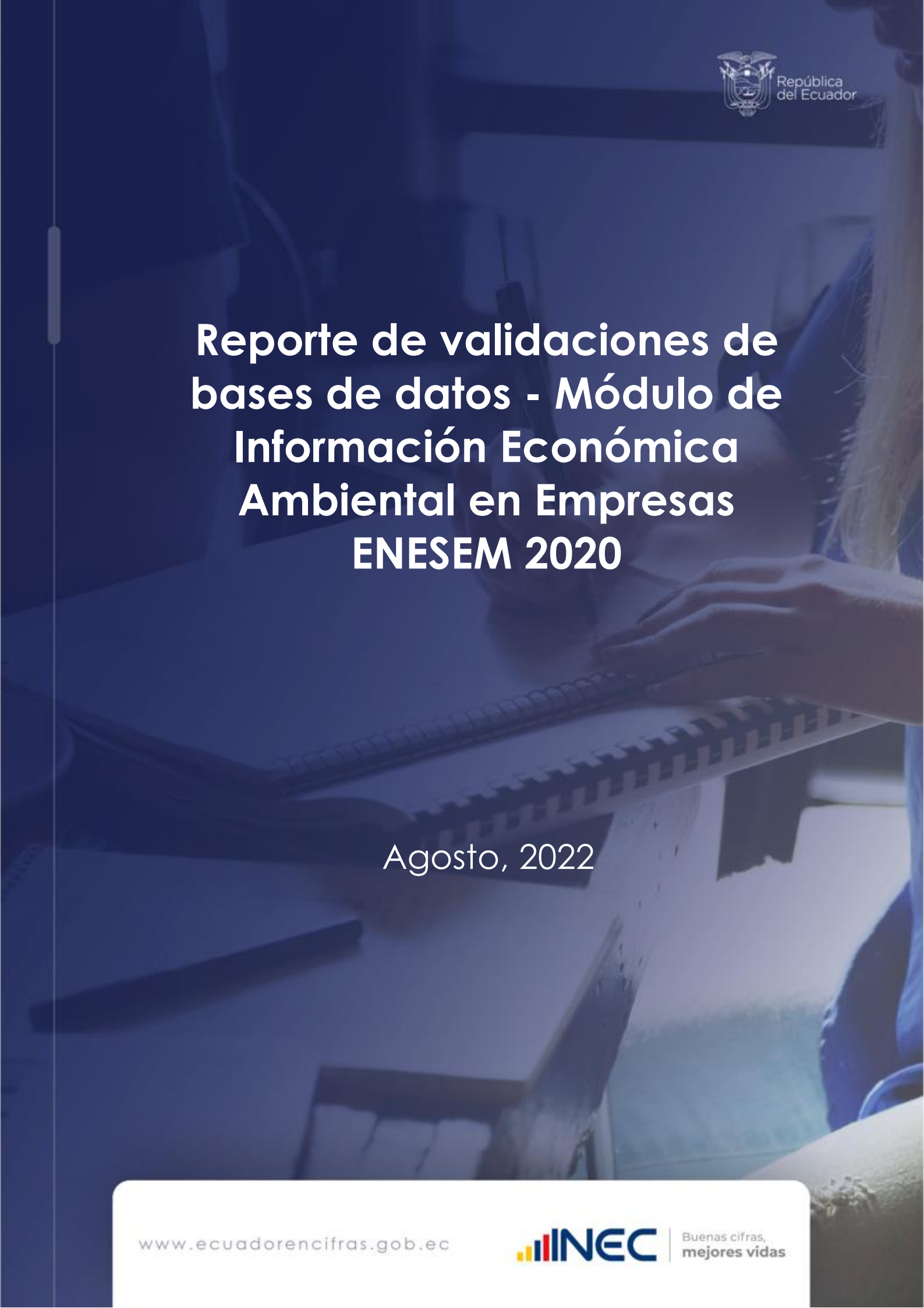

República del Ecuador

Reporte de validaciones de bases de datos - Módulo de Información Económica Ambiental en Empresas ENESEM 2020

Agosto, 2022

www.ecuadorencifras.gob.ec

INEC Buenas cifras, mejores vidas

Tabla de Contenido

Buenas cifras, mejores vidas

01.

Introducción

Introducción

Las operaciones estadísticas del Instituto Nacional de Estadística y Censos (INEC) siguen un modelo de producción estándar que sistematiza el quehacer estadístico a nivel de la institución, así como de todo el Sistema Estadístico Nacional (SEN). Se trata del **Modelo de Producción Estadística (MPE)**[1], que no es sino:

> "…*el conjunto de fases, procesos y actividades necesarias para producir estadísticas oficiales. A su vez, el modelo permite estandarizar y mejorar los datos y metadatos, y establecer una terminología común en el proceso de producción estadística entre las entidades del Sistema Estadístico Nacional*" (p. 5).

Según las directrices del MPE, existen 8 fases estándar del modelo (en este orden): Planificación, Diseño, Construcción, Recolección, Procesamiento, Análisis, Difusión y Evaluación. Estas fases serán aplicables a una determinada operación estadística, dependiendo de la naturaleza de la misma.

En el caso particular del Módulo de Información Ambiental Económica en Empresas (ENESEM), edición 2020, todas las fases designadas por el MPE se aplicaron a dicho módulo. En particular, en la fase de Procesamiento, existen 7 subprocesos que configuran esta fase: (1) Criticar e integrar la base de datos; (2) Clasificar y codificar; (3) Validar e imputar; (4) Derivar nuevas variables y unidades; (5) Ajustar los factores de expansión; (6) Tabular y generar indicadores; y (7) Finalizar los archivos de datos.

El presente reporte brinda información sobre las tareas y resultados obtenidos en el subproceso ***(3) Validar e imputar*** aplicados al Módulo de Información Ambiental Económica de la Encuesta Estructural Empresarial (ENESEM), edición 2020.

[1] INEC, 2016. Modelo de Producción Estadística del Ecuador. URL: https://www.ecuadorencifras.gob.ec/documentos/web-inec/Sistema_Estadistico_Nacional/Normativas_y_Estandares/Documento_del_Modelo_de_Produccion_Estadistica.pdf; accedido el 24 de agosto de 2022.

Buenas cifras, mejores vidas

02.

Fase de Procesamiento del MPE: tipos de validaciones e imputación de la BDD.

En la fase de Procesamiento del Modelo de Producción Estadística,

> "…*se ejecutan actividades de crítica, codificación, digitalización y validación de los datos recolectados, así como la generación de los tabulados e indicadores de la operación. También se llevan a cabo métodos para proteger y salvaguardar la seguridad de los datos en cada uno de los procesos de esta fase. Para los resultados estadísticos producidos regularmente, esta fase se produce en cada iteración*" (p. 18).

Todos los procesos y actividades de esta fase:

> "…*se ejecutarán acorde a las buenas prácticas de los principios de: i) Procedimientos Estadísticos Adecuados, ii) Precisión y Confiabilidad, y iii) Coherencia y Comparabilidad, contenidos en el Código de Buenas Prácticas Estadísticas*". (p. 19).

De toda la fase de Procesamiento del MPE, quizá el proceso de mayor importancia es el de ***Validación e Imputación***. Según el Modelo de Producción Estadística:

> "*Este proceso está encaminado al tratamiento de errores en la información y de ser necesario solventar la información faltante o datos perdidos. Aquí se toman en cuenta los criterios establecidos en el plan de validación y de imputación los cuales fueron elaborados en la fase de diseño (2.6 Diseñar el procesamiento y análisis)*". (p. 20).

Con respecto a las tareas que típicamente suelen realizarse en el proceso de Validación e Imputación, el Modelo de Producción Estadística prescribe que:

> "*En la validación se examinan los datos para tratar de identificar los posibles problemas, errores y discrepancias, como los valores atípicos, la falta de respuesta y el error de codificación. Este proceso se puede ejecutar de forma iterativa o mediante la validación de los datos contra reglas predefinidas en un orden establecido. La validación se aplica a los datos, independiente de su fuente u origen, antes y después de la integración*". (p.20).

El MPE menciona que algunas o todas las tareas del proceso de Validación e Imputación se pueden efectuar en paralelo, a lo largo del desarrollo de las actividades de la fase previa de Recolección de Datos.

Con respecto a la imputación de los datos incorrectos, no verificables o faltantes,

> "*En caso de que se requiera, se pueden emplear métodos de imputación de manera rigurosa, sustentada y documentada. Si los datos se consideran incorrectos o no fiables, se pueden remplazar por nuevos valores calculados a través de métodos estadísticos robustos y probados. Existe una variedad de métodos para imputar y a menudo se utiliza un enfoque basado en reglas*". (p. 20).

En el caso del Módulo de Información Ambiental Económica de la ENESEM 2020, se han implementado no solamente uno, sino varios tipos de validaciones con los cuales se intenta maximizar el grado de calidad de la información recabada en la fase de Recolección (Levantamiento en campo) de la Información. Los tipos de validaciones aplicados al Módulo de Información Ambiental Económica de la ENESEM 2020 son los siguientes:

1. ***Validación estándar (o lógica):*** Por la cual se valida la información de una variable con referencia a ciertos valores definidos como "válidos" de otras variables relacionadas con la variable a validar. Por ejemplo, si los valores de la variable **v7001** debe ser tal que iguale a la variable **v5090**, se crea una variable de control denominada **c7001**, en la cual se marca con "1" a las empresas que no cumplan con la condición lógica: **v7001 == v5090**. En el Anexo A se adjunta el código R que ejecuta este subproceso.
2. ***Control de integridad:*** Por la cual se ejecuta código corrector de valores que no pudieron ser validados en las diferentes rondas de validación, sea porque no se logró corroborar el dato en campo, sea porque la empresa investigada no quiso entregar el dato, o por cualquier otra razón. El código que realiza el control de integridad ejecuta, efectivamente, una imputación de datos. Sin embargo, esta imputación no aplica valores que pueden resultar de simulaciones o de aproximaciones estadísticas, sino que corresponden a valores finitos y determinados que deberían tener las variables con problemas de falta de integridad. En ese sentido, puede decirse que la imputación es *limitada* a ciertos

valores que debería exhibir una variable en condiciones normales. Por ejemplo, si la regla de validación lógica **c10ii6** determina que debe haber un valor entero entre 0 y 100 para el registro del porcentaje de aguas residuales tratadas (variable **v10ii6**), pero por efecto de algún borrado involuntario del crítico se perdió la información de la variable **v10ii6**, se recupera esta información de otras variables que generan el contexto de validación para la **v10ii6**, como es la variable **v10ii2** ("¿Los procesos productivos de la empresa generaron aguas residuales?"). Así, si la **v10ii2** registra código 2 (= "No"), entonces el porcentaje de aguas residuales tratadas es 0%, valor que debería ir en la **v10ii6**. Además, dado que hay otro grupo de variables cuyos valores dependen del valor de **v10ii6** (como son la secuencia de variables **v10036, …, v10048** que registran el destino de las aguas residuales tratadas; así como la secuencia de variables **v10049, …, v10061** que registran el destino de las aguas residuales no tratadas), se tiene que colocar los valores válidos correspondientes en estas otras variables (cuyo contexto se define parcial o totalmente con la variable bajo escrutinio **v10ii6**) para que todo el subsistema de variables sea coherente. En el Anexo B se adjunta el código R que ejecuta este subproceso.

Esta validación suele realizarse, generalmente, en la última ronda de validaciones, debido a que se dispone de la mayor muestra posible para estimar la distribución teórica de probabilidades de las variables y, por tanto, se disminuye la varianza y se minimiza la cantidad de "falsos positivos" que pudieran generar una carga de trabajo extra innecesaria para el personal de Crítica.

3. ***Validaciones especiales:*** Por las cuales se determina cuáles empresas no han ingresado la observación obligatoria debida al registro del código 3 ("Otros usos") para la energía eléctrica complementaria generada y consumida, así como para los combustibles y lubricantes utilizados. También abarca la revisión manual de observaciones no pertinentes a los otros usos de energía y combustibles. En el Anexo C se adjunta el código R que ejecuta este subproceso.
4. ***Validaciones de comparaciones de valores 2020 con valores 2019:*** En esta validación se suele calcular la variación interanual relativa (2019-2020) para todas las variables escalares del módulo. Luego, se procede a aplicar el logaritmo decimal a estas variaciones interanuales, con el fin de minimizar el efecto de "apreciación magnificada" que ocurre cuando existen variaciones en varios órdenes de magnitud. A los valores extremos superiores de esta

distribución logaritmada de variaciones interanuales se las marca para ser enviadas a Crítica, con el fin de comprobar si efectivamente la variación de valores de la variable bajo análisis es correcta, o si se deben corregir valores, o finalmente ratificar los altos niveles de variación interanual con una justificación válida. En el Anexo D se adjunta el código R que ejecuta este subproceso.

Esta validación suele realizarse, generalmente, en la última ronda de validaciones, debido a que se dispone de la mayor muestra posible para estimar la distribución teórica de probabilidades de las variables y, por tanto, se disminuye la varianza y se minimiza la cantidad de "falsos positivos" que pudieran generar una carga de trabajo extra innecesaria para el personal de Crítica.

5. ***Validaciones de verificación de valores extremos:*** Usualmente, las distribuciones de las variables escalares (crematísticas o de cantidad) suelen generar valores atípicos y extremos, tanto inferiores como superiores. Lo que se hace con este tipo de validación es fijar una variable de escala, y luego marcar a las empresas cuyos valores son inferiores al umbral inferior (límite válido inferior), así como a las que tiene valores superiores al umbral superior (límite válido superior). Estos casos tienen que verificarse por parte de los críticos de las zonales quienes, de ser necesario, corregirán los valores erróneos o ratificar los valores ingresados con la correspondiente justificación. En el Anexo E se adjunta el código R que ejecuta este subproceso.

   Esta validación suele realizarse, generalmente, en la última ronda de validaciones, debido a que se dispone de la mayor muestra posible para estimar la distribución teórica de probabilidades de las variables y, por tanto, se disminuye la varianza y se minimiza la cantidad de "falsos positivos" que pudieran generar una carga de trabajo extra innecesaria para el personal de Crítica.

Una vez descritos en detalle los diferentes métodos o tipos de validaciones aplicados a los datos recolectados para las variables del Módulo de Información Ambiental Económica de la ENESEM 2020, se procede a reportar los resultados encontrados para estas validaciones en los diferentes cortes de la base de datos, correspondientes a las 11 rondas de validación efectuadas.

Buenas cifras, mejores vidas

03.

Reporte de resultados de validaciones por ronda.

Los resultados de las validaciones lógicas, según los 11 cortes de las bases de datos correspondientes a las rondas de validación planificadas, fueron las siguientes:

1. **Corte #1 al 02-08-2021:** De un universo de 191 empresas efectivas y criticadas, 180 empresas tuvieron al menos un error de validación lógica. Estas 180 empresas se distribuyen por zonal como: 18=Zonal Centro; 110=Zonal DICA; 29=Zonal Litoral; 23=Zonal Sur. Entre todas las zonales, se generaron 40 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control[2].
2. **Corte #2 al 16-08-2021:** De un universo de 503 empresas efectivas y criticadas, 99 empresas[3] tuvieron al menos un error de validación lógica. Estas 99 empresas se distribuyen por zonal como: 26=Zonal Centro; 43=Zonal DICA; 12=Zonal Litoral; 18=Zonal Sur. Entre todas las zonales, se generaron 159 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
3. **Corte #3 al 30-08-2021:** De un universo de 783 empresas efectivas y criticadas, 33 empresas tuvieron al menos un error de validación lógica. Estas 33 empresas se distribuyen por zonal como: 3=Zonal Centro; 18=Zonal DICA; 5=Zonal Litoral; 7=Zonal Sur. Entre todas las zonales, se generaron 78 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
4. **Corte #4 al 17-09-2021:** De un universo de 2842 empresas efectivas y criticadas, 483 empresas tuvieron al menos un error de validación lógica. Estas 484 empresas se distribuyen por zonal como: 23=Zonal Centro; 135=Zonal DICA; 279=Zonal Litoral; 46=Zonal Sur. Entre todas las zonales, se generaron 222 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
5. **Corte #5 al 30-09-2021:** De un universo de 3387 empresas efectivas y criticadas, 58 empresas tuvieron al menos un error de validación lógica. Estas 58 empresas se distribuyen por zonal como: 3=Zonal Centro; 17=Zonal DICA; 31=Zonal Litoral; 7=Zonal Sur. Entre todas las zonales, se generaron 75 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.

[2] Correspondientes a 1188 reglas de validación lógica de toda la malla de validación lógica.

[3] Estas empresas no incluyen a ninguna de las existentes en el 1er corte. Esto equivale a asumir que todos los errores de las empresas del 1er corte fueron corregidos o justificados con observaciones de crítica, lo cual no es totalmente cierto. Sin embargo, todos los errores de validación lógica que quedaron sin corregir o justificar durante las primeras 10 rondas de validación quedaron para ser resueltos en la 11ra y última jornada de validación.

6. **Corte #6 al 15-10-2021:** De un universo de 3760 empresas efectivas y criticadas, 107 empresas tuvieron al menos un error de validación lógica. Estas 107 empresas se distribuyen por zonal como: 4=Zonal Centro; 40=Zonal DICA; 43=Zonal Litoral; 20=Zonal Sur. Entre todas las zonales, se generaron 60 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
7. **Corte #7 al 25-10-2021:** De un universo de 3967 empresas efectivas y criticadas, 45 empresas tuvieron al menos un error de validación lógica. Estas 45 empresas se distribuyen por zonal como: 5=Zonal Centro; 19=Zonal DICA; 7=Zonal Litoral; 14=Zonal Sur. Entre todas las zonales, se generaron 44 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
8. **Corte #8 al 05-11-2021:** De un universo de 4052 empresas efectivas y criticadas, 16 empresas tuvieron al menos un error de validación lógica. Estas 16 empresas se distribuyen por zonal como: 1=Zonal Centro; 10=Zonal DICA; 2=Zonal Litoral; 3=Zonal Sur. Entre todas las zonales, se generaron 31 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
9. **Corte #9 al 25-11-2021:** De un universo de 4058 empresas efectivas y criticadas, 36 empresas tuvieron al menos un error de validación lógica. Estas 16 empresas se distribuyen por zonal como: 0=Zonal Centro; 28=Zonal DICA; 6=Zonal Litoral; 2=Zonal Sur. Entre todas las zonales, se generaron 26 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.
10. **Corte #10 al 02-12-2021:** Debido a que el universo de empresas se contrajo a 3127 empresas efectivas y criticadas, no se procedió a realizar ninguna validación en esta ronda.
11. **Corte #11 al 07-03-2022:** De un universo de 4065 empresas efectivas y criticadas[4], 440 empresas tuvieron al menos un error de validación lógica. Estas 440 empresas se distribuyen por zonal como: 59=Zonal Centro; 304=Zonal DICA; 64=Zonal Litoral; 13=Zonal Sur. Hay que notar que algunas de estas empresas son rezagos de rondas de validación anteriores. Entre todas las zonales, se generaron 186 variables de control con al menos un error a verificar, de un total teórico de 1188 variables de control.

[4] Hubo 13 empresas que se les declaró como "no efectivas" según el punto de vista de la falta de información económica, a pesar de que 12 de ellas tenían mucha información ambiental. Por tal razón, las empresas efectivas y criticadas finales, desde el punto de vista de la información ambiental, se redujeron a 4052.

Hay que resaltar el hecho que, finalmente, quedaron 3777 empresas en calidad de empresas publicables. Según manifestaron técnicos de DINEM, esto se debe a que hubo mucha rotación en los dominios de la muestra, entendiendo a cada "dominio" como el cruce de cada una de las 17 actividades económicas investigadas con los tres tamaños de empresa investigados (dando un total de 51 dominios). Por tal motivo, muchas empresas efectivas y criticadas del 11er corte salieron de la definición y optimización de los factores de expansión.

Con respecto a la validación de integridad, se ejecutó el código R que logró corregir los valores de 28 variables (de un total de 1294 variables del Módulo de Información Ambiental Económica de la ENESEM 2020), afectando a 641 empresas de un total de 4052 empresas efectivas (con información ambiental válida) y criticadas. En total, se cambiaron 10,401 valores de un total de 5,243,288 valores recolectados en campo (= 0.1984%).

En el tema de las validaciones especiales, normalmente se encontraban entre 0 y N, donde N es un valor menor que 16 empresas, para enviar a revisión a las zonales en cada ronda de validación. Este tipo de validación afecta solamente a algunas variables de generación de energía complementaria y su uso principal, así como las de generación de combustibles y su uso principal.

Los resultados de las validaciones por comparación de valores entre los años 2019 y 2020, según los 11 cortes de las bases de datos correspondientes a las rondas de validación planificadas, fueron las siguientes:

1. **Corte #1 al 02-08-2021:** De un universo de 189 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 160 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido[5]. Estas 160 empresas se distribuyen por zonal como: 30=Zonal Centro; 89=Zonal DICA; 24=Zonal Litoral; 17=Zonal Sur. Entre todas las zonales, se generaron 233 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
2. **Corte #2 al 16-08-2021:** De un universo de 513 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 394

[5] Este umbral fue fijado en 30%, después de hacer análisis de distribuciones de variación interanual de todas las variables escalares del Módulo Ambiental de la ENESEM 2020.

empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 394 empresas se distribuyen por zonal como: 90=Zonal Centro; 179=Zonal DICA; 80=Zonal Litoral; 45=Zonal Sur. Entre todas las zonales, se generaron 208 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.

3. **Corte #3 al 30-08-2021:** De un universo de 596 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 194 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 194 empresas se distribuyen por zonal como: 27=Zonal Centro; 113=Zonal DICA; 26=Zonal Litoral; 28=Zonal Sur. Entre todas las zonales, se generaron 59 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
4. **Corte #4 al 17-09-2021:** De un universo de 2323 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 1961 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 1961 empresas se distribuyen por zonal como: 104=Zonal Centro; 689=Zonal DICA; 986=Zonal Litoral; 182=Zonal Sur. Entre todas las zonales, se generaron 66 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
5. **Corte #5 al 30-09-2021:** De un universo de 2215 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 323 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 323 empresas se distribuyen por zonal como: 10=Zonal Centro; 102=Zonal DICA; 178=Zonal Litoral; 33=Zonal Sur. Entre todas las zonales, se generaron 58 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
6. **Corte #6 al 15-10-2021:** De un universo de 2703 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 2208 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 2208 empresas se distribuyen por zonal como: 112=Zonal Centro; 746=Zonal DICA; 1148=Zonal Litoral; 202=Zonal Sur. Entre todas las zonales, se generaron 66 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
7. **Corte #7 al 25-10-2021:** De un universo de 3097 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 2856 empresas tuvieron al menos una variable con una variación interanual mayor al

umbral establecido. Estas 2856 empresas se distribuyen por zonal como: 172=Zonal Centro; 1053=Zonal DICA; 1308=Zonal Litoral; 323=Zonal Sur. Entre todas las zonales, se generaron 66 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.

8. **Corte #8 al 05-11-2021:** De un universo de 3118 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 2925 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 2925 empresas se distribuyen por zonal como: 185=Zonal Centro; 1108=Zonal DICA; 1307=Zonal Litoral; 325=Zonal Sur. Entre todas las zonales, se generaron 56 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
9. **Corte #9 al 25-11-2021:** De un universo de 3120 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 1800 empresas tuvieron al menos una variable con una variación interanual mayor al umbral establecido. Estas 1800 empresas se distribuyen por zonal como: 88=Zonal Centro; 695=Zonal DICA; 788=Zonal Litoral; 229=Zonal Sur. Entre todas las zonales, se generaron 66 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.
10. **Corte #10 al 02-12-2021:** Debido a que el universo de empresas pasó de 3120 a a 3127 empresas efectivas y criticadas, no se procedió a realizar ninguna validación en esta ronda. Se decidió esperar a la última ronda de validación para ejecutar los algoritmos de comparación de variaciones interanuales.
11. **Corte #11 al 07-03-2022:** De un universo de 3647 empresas efectivas y criticadas que resultan de la intersección entre las BDD 2020 y 2019 (publicada), 225 empresas tuvieron al menos un error de validación lógica. Estas 225 empresas se distribuyen por zonal como: 13=Zonal Centro; 104=Zonal DICA; 91=Zonal Litoral; 17=Zonal Sur. Hay que notar que casi todas de estas empresas son rezagos de rondas de validación anteriores. Entre todas las zonales, se generaron 14 variables de control con al menos un valor al umbral permitido, de un total teórico de 741 variables de control.

Finalmente, con respecto a la verificación de valores extremos efectuada sobre el último corte al 07 de marzo de 2022, se obtuvo los siguientes resultados:

- Se tomó como universo a las 4049 empresas[6] validadas y criticadas hasta la 11ra ronda de validación. El universo de variables a controlar abarca a 681 variables escalares, de las cuales 150 miden porcentajes y 531 son variables de cantidades como ingresos, gastos, cantidad de energía, cantidad de residuos generada, etc. Se detectaron 509 empresas con al menos una de las 531 variables a controlar con al menos un valor extremo (sea supremo o ínfimo) de las distribuciones de valores de estas variables, segmentadas por el tamaño de empresa[7]. Se obtuvieron los siguientes resultados por zonal:

    - Zonal Centro: Hubo 32 empresas de 262 publicables de esta zonal con al menos un valor extremo a ser revisado. Se generaron 23 supremos, 21 ínfimos y 44 extremos en total. Se detectaron valores extremos en 22 de las 531 variables escalares a validar.
    - Zonal DICA: Hubo 189 empresas de 1524 publicables de esta zonal con al menos un valor extremo a ser revisado. Se generaron 113 supremos, 162 ínfimos y 275 extremos en total. Se detectaron valores extremos en 33 de las 531 variables escalares a validar.
    - Zonal Litoral: Hubo 220 empresas de 1810 publicables de esta zonal con al menos un valor extremo a ser revisado. Se generaron 136 supremos, 154 ínfimos y 290 extremos en total. Se detectaron valores extremos en 33 de las 531 variables escalares a validar.
    - Zonal Sur: Hubo 68 empresas de 453 publicables de esta zonal con al menos un valor extremo a ser revisado. Se generaron 54 supremos, 62 ínfimos y 116 extremos en total. Se detectaron valores extremos en 32 de las 531 variables escalares a validar.

Cabe mencionar que todas estas 509 empresas de las 4049 con potencial de ser publicables fueron justificadas en su totalidad sobre sus valores extremos. Aproximadamente el 72% de valores a verificar fueron corregidos, sea porque

[6] De las 4052 empresas que se dieron para el 11er corte de la base de datos, se eliminan 3 empresas para la aplicación del algoritmo de verificación de valores extremos porque se detectó que eran empresas fusionadas o absorbidas durante el levantamiento de campo.

[7] Esto implica que, por ejemplo, para la variable ***v7001*** no existe únicamente una distribución de los valores logaritmados de esa variable, sino tres: uno para c/u de los tamaños de empresa (Mediana A, Mediana B y Grande). Si una empresa sobrepasa los umbrales por tamaño, determinados por el algoritmo de la función del lenguaje R **robustbase::adjbox()**, entonces se la marca con el código "SUP" para indicar que el valor extremo detectado es un supremo de la distribución logaritmada. Si se lo marca con el código "INF", entonces se la marca con el código "INF" para indicar que el valor extremo detectado es un ínfimo de la distribución logaritmada.

fueron demasiado bajos, sea porque fueron demasiado altos con respecto a los valores de las empresas similares por tamaño. El resto fueron corroborados por crítica y/o campo.

Buenas cifras,
mejores vidas

04.

Conclusiones.

Como se ha mencionado anteriormente, la fase de Validación e Imputación del Modelo de Producción Estadística es crucial para el aseguramiento de la calidad de la información recolectada en campo para una operación estadística.

En el caso de la fase de Validación e Imputación del Módulo de Información Ambiental Económica de la ENESEM 2020, se la ha implementado desde varias perspectivas, cada una de las cuales minimiza la cantidad de errores de levantamiento, garantizando así la mejor calidad de la información disponible para los usuarios de esta operación estadística.

La implantación continua de los cinco tipos de validación que se utilizan para validar el Módulo de Información Ambiental Económica de la ENESEM 2020 presenta un alto grado de innovación, que se manifiesta a través de la automatización casi total de las tareas y actividades. Para este fin, se ha utilizado la plataforma de programación RStudio 4.1.2, en la cual se han desarrollado varios archivos o módulos de código R en su versión "R Markdown", la cual es ejecutable por bloques de código según la necesidad del programador.

Los resultados obtenidos confirman la tendencia creciente hacia la mejora continua de la calidad de la información generada en la presente operación estadística. De esta manera, el INEC cumple con su misión de garantizar a los usuarios especializados y al público en general el acceso a información robusta, oportuna y accesible universalmente para apoyar al desarrollo efectivo de los sectores productivos del país.

Buenas cifras,
mejores vidas

05.

Anexos.

ANEXO A. Código R Markdown para la validación estándar o lógica.

```
---
title: "Validacion Lógica ENESEM 2020"
author: "Ing. Ramiro Benavides L."
date: "28/08/2020"
update: "17/08/2021"
output:
  html_document:
    df_print: paged
  html_notebook: default
  pdf_document: default
---
```

A. Carga de la base de datos en formato CSV. Se simula la variable "Zonal_DEAGA", en caso de ser necesario.

```
```{r}
# CSV <- file.choose()
# B <- read.csv(CSV)
# if (!("Zonal_DEAGA" %in% names(B))) {
#   w <- B$inec_zonal
#   T1 <- data.frame(B, w)
#   B <- T1
#   rm(T1, w)
#   names(B)[dim(B)[2]] <- paste("Zonal_DEAGA")
# }
# library(haven)
# EmpresasPreliminar_16_11_2020 <- read_sav("C:/Users/Ramiro/Desktop/EMPRESAS 2019/3er Corte - 16-11-
2020/EmpresasPreliminar_16_11_2020.sav")
# detach("package:haven", unload = TRUE)

EMP_2020 <- haven::read_sav("C:/Users/Ramiro/Documents/EMPRESAS 2020/10mo Corte - 02-12-
2021/BDD/Empresasfusionado.sav")

B <- EMP_2020[which(EMP_2020$estado == "C" & EMP_2020$efectividad == 1), ]
# B <- EMP_2020[!is.na(EMP_2020$nombre_critico), ]

# Eliminar primero las empresas no efectivas (dejar únicamente a las efectivas).
B <- B[B$efectividad==1, ]

unique(B$nombre_critico)

# Eliminar a las empresas fusionadas y absorbidas.
B <- B[B$fusionada!="1.7" & B$absorbida!="1.8", ]

# Eliminar a la empresa TAME, que es efectiva y criticada, pero ya no existe.
B <- B[-which(B$inec_identificador_empresa == "41909487178"), ]
```


```{r}
# Asignar empresas a zonal, según el nombre del crítico.
B$Zonal_DEAGA <- NA_character_
B$Zonal_DEAGA[B$nombre_critico == "MUÑOZ MORA HARLETH ENYTH"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "CASTRO VILLALVA ROBERTO MAURICIO"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "SUMBA AYALA MARCIA IVON"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "AYALA SUMBA  MARCIA IVON"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "TERAN LOJA ADRIAN OSWALDO"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "VERGARA URGILES  REBECA"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "MALIZA CHASI LUZ ANGELICA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "YAUTIBUG BARRERA KATERIN PAOLA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "URGILÉS URGILÉS ENMA CUMANDÁ"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SANCHEZ RAMIREZ GERMAN ANDRES"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "MORETA SARAGURO EVELYN SOFIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "GONZALEZ PIONCE FABRIZIO ADRIAN"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "VALENCIA PEREZ FERNANDO DANIEL"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "BEDOYA PEÑAFIEL ANDRES PATRICIO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "BALSECA VILLALVA LESLIE PRISCILLA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "HOLGUIN QUIJIJE KENYS ELIZABETH"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "NIETO VERA MARYLIN KATHERINE"] <- "Litoral"
```
```

```
B$Zonal_DEAGA[B$nombre_critico == "VILLACIS VILLACIS VICTOR HUGO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "PEÑA ICAZA ARMANDO BERNARDINO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "BENITES CAMPOVERDE JESSICA DENISE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "RAMOS MUÑOZ JIREMY JACQUELINE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "ALVARADO CEDEÑO SELENE GARDENIA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "NACEVILLA CACHAGO MIREYA PATRICIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "GUEVARA ALVARADO ELICEO FRANCISCO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "GOMEZ MEJIA BYRON DAVID"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "VALDEZ SALAZAR JOSSELIN NOHELY"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SEVILLA VILLACIS CRISTINA SOLANGE"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "HERNANDEZ JATIVA MARIA PAULINA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "PAREDES ROMERO ALEXANDER JAVIER"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "CALIXTO FARIÑO JOHNNY ALEXIS"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "SEMANATE CAJIAO ANTONIO JAVIER"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SEMANATE ALVAREZ LEONELA DE LOS ANGELES"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "TOMALA MIRANDA STEFANIE YESENIA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "RODRIGUEZ BENAVIDES BRAYAN VLADIMIR"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "RUANO VACA CRISTINA ELIZABETH"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "CALLE PALACIOS MAGALY CARLOTA"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "GUARDERAS NÚÑEZ TATIANA VALERIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "PICO SANCHEZ ROSA PATRICIA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "MAIGUA VELA MAGDALENA DEL CONSUELO"] <- "DICA"

# B$Zonal_DEAGA[which(is.na(B$Zonal_DEAGA) & B$efectividad == 1 & B$inec_zonal == 2)] <- "Centro"
# B$Zonal_DEAGA[which(is.na(B$Zonal_DEAGA) & B$efectividad == 1 & B$inec_zonal == 3)] <- "Litoral"
# B$Zonal_DEAGA[which(is.na(B$Zonal_DEAGA) & B$efectividad == 1 & B$inec_zonal == 4)] <- "DICA"
# B$Zonal_DEAGA[which(is.na(B$Zonal_DEAGA) & B$efectividad == 1 & B$inec_zonal == 5)] <- "Sur"

# B$Zonal_DEAGA[B$nombre_critico=="ANCHALUISA BARONA ROCIO ELIZABETH"] <- "SIN_ZONAL_1"
# B$Zonal_DEAGA[B$nombre_critico=="BUCHELI VASQUEZ IRENE ABIGAIL"] <- "SIN_ZONAL_2"
# B$Zonal_DEAGA[B$nombre_critico=="ALARCÓN SOLORZANO EDISON SANTIAGO"] <- "SIN_ZONAL_3"
# B$Zonal_DEAGA[B$nombre_critico=="BORJA SISALEMA CARLA ESTEFANÍA"] <- "SIN_ZONAL_4"
# B$Zonal_DEAGA[B$nombre_critico=="CAÑIZARES RODRIGUEZ MONICA CECILIA"] <- "SIN_ZONAL_5"
# B$Zonal_DEAGA[B$nombre_critico=="GUERRA CAMINO JAVIER ALEJANDRO"] <- "SIN_ZONAL_6"
# B$Zonal_DEAGA[B$nombre_critico=="CHIMBORAZO GUAMAN MARIA ROSA"] <- "SIN_ZONAL_7"
# B$Zonal_DEAGA[B$nombre_critico=="CEVALLOS TIPAN FRANCISCO JAVIER"] <- "SIN_ZONAL_8"
# B$Zonal_DEAGA[2833] <- NA  # Empresa fusionada.
# B$efectividad[which(is.na(B$efectividad) & !is.na(B$Zonal_DEAGA))] <- 1 # Empresas
# # sin efectividad asignadas, pero con datos en las variables económicas y ambientales.
# B <- B[which(B$efectividad==1 & !is.na(B$Zonal_DEAGA)), ]
# B_orig <- B
# B <- B[-grep("^SIN", B$Zonal_DEAGA), ]
```

```{r}
# Si es necesario, crear las columnas B$UNO y B$CIEN.
B$UNO <- 1
B$CIEN <- 100
# Esta línea se corrió en la 5ta ronda de validaciones. Se declara a esta empresa NO EFECTIVA
# por falta de información.
# B <- B[-which(B$secuencial_sistema == 44408), ]
# B <- B[-which(B$secuencial_sistema == 45366), ]
```

B. Creación de filtro "Efectivas" para filtrar los casos con errores.

Este filtrado se hará después de finalizar la corrida de validaciones, sobre todo en el archivo CSV que se debe enviar a las zonales. Sin embargo, el filtrado se lo hará en R, y no en Excel.

```{r}
# Efectivas <- ifelse(B$efectividad == 1, 1, 0)
# W <- (Efectivas == 1)
# B <- B[-which(is.na(B$v8002)), ]
```

C. Crear el dataframe con las variables de identificación, el cual irá agregando después las variables de validación. Este frame se enviará a las zonales para su validación.

```{r}
```

```
lista_vars_identif <- c("inec_zonal", "Zonal_DEAGA", "inec_tamanio", "inec_ciiu4", "inec_identificador_empresa", "novedad",
                "efectividad", "ruc_muestra", "ruc", "nombre_comercial", "razon_social", "desc_actividad_principal",
                "ciiu4_actividad_principal", "desc_actividad_secundaria", "ciiu4_actividad_secundaria", "nombre_critico",
"fecha_critica")
G <- subset(B, select = lista_vars_identif)
G_orig <- G
```

D. Definición de funciones utilitarias.

```{r}
# Función que calcula la resta de una columna base con otras columnas de un dataframe.
CalcDiffCol <- function(colbase, col1, ...) {
  y <- apply(cbind(col1, ...), 1, sum, na.rm = T)
  z <- apply(cbind(-y, colbase), 1, sum, na.rm = T)
  if (sum(z, na.rm = T) != 0) x <- ifelse(z != 0, 1, NA)
}

# Función de agregación de reglas de validación únicamente de tipo control de sumas.
AddValidSumas <- function(label, colbase, col1, ..., filtro="") {
  p <- CalcDiffCol(colbase, col1, ...)
  if (!(is.null(p))) {
    comando1 <- paste0("K <- dim(B)[1]")
    comando2 <- paste0("K <- length(which(", filtro, "))")
    comando <- ifelse(filtro == "", comando1, comando2)
    eval(parse(text = comando), envir = .GlobalEnv)
    M <- get0("K")
    r <- get0("G")
    comando1 <- paste0("r$", label, " <- p")
    comando2 <- paste0("r$", label, " <- ifelse(", filtro, ", p, NA)")
    comando <- ifelse(filtro == "", comando1, comando2)
    eval(parse(text = comando))
    if (!(all(is.na(r[dim(r)[2]])))) {
      assign("G", r, envir = .GlobalEnv)
      N <- sum(r[dim(r)[2]], na.rm = T)
      print(paste0("SÍ se agregó la regla ", label, " al frame de validación. ",
                   "N/M=", N, "/", M))
    } else {
      print(paste0("NO se agregó la regla ", label, " al frame de validación"))
    }
    comando <- paste0("rm(K)")
    eval(parse(text = comando), envir = .GlobalEnv)
  } else {
    print(paste0("NO se agregó la regla ", label, " al frame de validación"))
  }
}

# Función de agregación de reglas de validación. Genera una variable de control de nombre “label”, aplicando una regla de
validación expresada por la cadena de texto “regla” a todas las empresas que cumplen con la restricción “filtro”.
AddRule <- function(label, regla, filtro="") {
  comando1 <- paste0("K <- dim(B)[1]")
  comando2 <- paste0("K <- length(which(", filtro, "))")
  comando <- ifelse(filtro == "", comando1, comando2)
  eval(parse(text = comando), envir = .GlobalEnv)
  M <- get0("K")
  r <- get0("G")
  comando1 <- paste0("r$", label, " <- ifelse(!(", regla, "), 1, NA)")
  comando2 <- paste0("r$", label, " <- ifelse(!(", regla, ") & (", filtro, "), 1, NA)")
  comando <- ifelse(filtro == "", comando1, comando2)
  eval(parse(text = comando))
  if (!(all(is.na(r[dim(r)[2]])))) {
     assign("G", r, envir = .GlobalEnv)
     N <- sum(r[dim(r)[2]], na.rm = T)
     print(paste0("SÍ se agregó la regla ", label, " al frame de validación. ",
                  "N/M=", N, "/", M))
  } else {
    print(paste0("NO se agregó la regla ", label, " al frame de validación"))
  }
```

```
  comando <- paste0("rm(K)")
  eval(parse(text = comando), envir = .GlobalEnv)
}
```

E. Validación de sumas: Capítulo 7

```{r}
AddValidSumas("s7002", B$v7002, B$v7003, B$v7004)
AddValidSumas("s7026", B$v7026, B$v7010, B$v7015, B$v7020, filtro="B$v742 > 0")
AddValidSumas("s7027", B$v7027, B$v7011, B$v7016, B$v7021, filtro="B$v742 > 0")
AddValidSumas("s7028", B$v7028, B$v7012, B$v7017, B$v7022, filtro="B$v742 > 0")
AddValidSumas("s7029", B$v7029, B$v7013, B$v7018, B$v7023, filtro="B$v742 > 0")
AddValidSumas("s7030a", B$v7030, B$v7014, B$v7019, B$v7024, filtro="B$v742 > 0")
AddValidSumas("s7014", B$v7014, B$v7010, B$v7011, B$v7012, B$v7013, filtro="B$v742 > 0")
AddValidSumas("s7019", B$v7019, B$v7015, B$v7016, B$v7017, B$v7018, filtro="B$v742 > 0")
AddValidSumas("s7024", B$v7024, B$v7020, B$v7021, B$v7022, B$v7023, filtro="B$v742 > 0")
AddValidSumas("s7030b", B$v7030, B$v7026, B$v7027, B$v7028, B$v7029, filtro="B$v742 > 0")
```

E. Validación de sumas: Capítulo 8

```{r}
AddValidSumas("s8098", B$v8098, B$v8003, B$v8009, B$v8015, B$v8021, B$v8027, B$v8033, B$v8039, B$v8045, B$v8051,
B$v8057, B$v8063, B$v8069, B$v8075, B$v8081, B$v8087, B$v8093)
AddValidSumas("s8099", B$v8099, B$v8005, B$v8011, B$v8017, B$v8023, B$v8029, B$v8035, B$v8041, B$v8047, B$v8053,
B$v8059, B$v8065, B$v8071, B$v8077, B$v8083, B$v8089, B$v8095)
AddValidSumas("s8100", B$v8100, B$v8007, B$v8013, B$v8019, B$v8025, B$v8031, B$v8037, B$v8043, B$v8049, B$v8055,
B$v8061, B$v8067, B$v8073, B$v8079, B$v8085, B$v8091, B$v8097)
```

E. Validación de sumas: Capítulo 9

```{r}
AddValidSumas("s9052", B$v9052, B$v9005, B$v9013, B$v9021, B$v9029, B$v9037, B$v9045, filtro="B$v9i2==1")
AddValidSumas("s9053", B$v9053, B$v9006, B$v9014, B$v9022, B$v9030, B$v9038, B$v9046, filtro="B$v9i2==1")
AddValidSumas("s9054", B$v9054, B$v9007, B$v9015, B$v9023, B$v9031, B$v9039, B$v9047, filtro="B$v9i2==1")
AddValidSumas("s9055", B$v9055, B$v9009, B$v9017, B$v9025, B$v9033, B$v9041, B$v9049, filtro="B$v9i2==1")
AddValidSumas("s9056", B$v9056, B$v9010, B$v9018, B$v9026, B$v9034, B$v9042, B$v9050, filtro="B$v9i2==1")
AddValidSumas("s9105", B$v9105, B$v9059, B$v9063, B$v9067, B$v9071, B$v9075, B$v9079,
        B$v9083, B$v9087, B$v9091, B$v9095, B$v9099, B$v9102, filtro="B$v9ii1==1")
```

E. Validación de sumas: Capítulo 10. Aguas de captación y residuales.

```{r}
AddValidSumas("s10030", B$v10030, B$v10012, B$v10020, B$v10028, filtro="B$v10i3==1")
AddValidSumas("s10031", B$v10031, B$v10013, B$v10021, B$v10029, filtro="B$v10i3==1")
AddValidSumas("s10048", B$v10048, B$v10037, B$v10039, B$v10041, B$v10043,
        B$v10045, B$v10047, filtro="B$v10ii6 %in% 1:100")
AddValidSumas("s10037", B$CIEN, B$v10037, B$v10039, B$v10041, B$v10043,
        B$v10045, B$v10047, filtro="B$v10ii6 %in% 1:100")
AddValidSumas("s10061", B$v10061, B$v10050, B$v10052, B$v10054, B$v10056, B$v10058,
        B$v10060, filtro="B$v10ii6 %in% 0:99")
AddValidSumas("s10050", B$CIEN, B$v10050, B$v10052, B$v10054, B$v10056, B$v10058,
        B$v10060, filtro="B$v10ii6 %in% 0:99")
```

E. Validación de sumas: Capítulo 10.1 Residuos No Peligrosos.

```{r}
AddValidSumas("s10074", B$v10074, B$v10067, B$v10069, B$v10071, B$v10073, filtro="B$v10062==1")
AddValidSumas("s10095", B$v10095, B$v10088, B$v10090, B$v10092, B$v10094, filtro="B$v10083==1")
AddValidSumas("s10116", B$v10116, B$v10109, B$v10111, B$v10113, B$v10115, filtro="B$v10104==1")
AddValidSumas("s10137", B$v10137, B$v10130, B$v10132, B$v10134, B$v10136, filtro="B$v10125==1")
AddValidSumas("s10158", B$v10158, B$v10151, B$v10153, B$v10155, B$v10157, filtro="B$v10146==1")
AddValidSumas("s10179", B$v10179, B$v10172, B$v10174, B$v10176, B$v10178, filtro="B$v10167==1")
AddValidSumas("s10200", B$v10200, B$v10193, B$v10195, B$v10197, B$v10199, filtro="B$v10188==1")
AddValidSumas("s10221", B$v10221, B$v10214, B$v10216, B$v10218, B$v10220, filtro="B$v10209==1")
AddValidSumas("s10242", B$v10242, B$v10235, B$v10237, B$v10239, B$v10241, filtro="B$v10230==1")
AddValidSumas("s10263", B$v10263, B$v10256, B$v10258, B$v10260, B$v10262, filtro="B$v10251==1")
AddValidSumas("s10284", B$v10284, B$v10277, B$v10279, B$v10281, B$v10283, filtro="B$v10272==1")
```

```
AddValidSumas("s10305", B$v10305, B$v10298, B$v10300, B$v10302, B$v10304, filtro="B$v10293==1")
AddValidSumas("s10326", B$v10326, B$v10319, B$v10321, B$v10323, B$v10325, filtro="B$v10314==1")
AddValidSumas("s10347", B$v10347, B$v10340, B$v10342, B$v10344, B$v10346, filtro="B$v10335==1")
AddValidSumas("s10368", B$v10368, B$v10361, B$v10363, B$v10365, B$v10367, filtro="B$v10356==1")
AddValidSumas("s10079", B$CIEN, B$v10079, B$v10080, B$v10081, filtro="B$v10072==1")
AddValidSumas("s10100", B$CIEN, B$v10100, B$v10101, B$v10102, filtro="B$v10093==1")
AddValidSumas("s10121", B$CIEN, B$v10121, B$v10122, B$v10123, filtro="B$v10114==1")
AddValidSumas("s10142", B$CIEN, B$v10142, B$v10143, B$v10144, filtro="B$v10135==1")
AddValidSumas("s10163", B$CIEN, B$v10163, B$v10164, B$v10165, filtro="B$v10156==1")
AddValidSumas("s10184", B$CIEN, B$v10184, B$v10185, B$v10186, filtro="B$v10177==1")
AddValidSumas("s10205", B$CIEN, B$v10205, B$v10206, B$v10207, filtro="B$v10198==1")
AddValidSumas("s10226", B$CIEN, B$v10226, B$v10227, B$v10228, filtro="B$v10219==1")
AddValidSumas("s10247", B$CIEN, B$v10247, B$v10248, B$v10249, filtro="B$v10240==1")
AddValidSumas("s10268", B$CIEN, B$v10268, B$v10269, B$v10270, filtro="B$v10261==1")
AddValidSumas("s10289", B$CIEN, B$v10289, B$v10290, B$v10291, filtro="B$v10282==1")
AddValidSumas("s10310", B$CIEN, B$v10310, B$v10311, B$v10312, filtro="B$v10303==1")
AddValidSumas("s10331", B$CIEN, B$v10331, B$v10332, B$v10333, filtro="B$v10324==1")
AddValidSumas("s10352", B$CIEN, B$v10352, B$v10353, B$v10354, filtro="B$v10345==1")
AddValidSumas("s10373", B$CIEN, B$v10373, B$v10374, B$v10375, filtro="B$v10366==1")
```

E. Validación de sumas: Capítulo 10.2 Desechos especiales.

```{r}
AddValidSumas("s10389", B$v10389, B$v10382, B$v10384, B$v10386, B$v10388, filtro="B$v10377==1")
AddValidSumas("s10410", B$v10410, B$v10403, B$v10405, B$v10407, B$v10409, filtro="B$v10398==1")
AddValidSumas("s10431", B$v10431, B$v10424, B$v10426, B$v10428, B$v10430, filtro="B$v10419==1")
AddValidSumas("s10452", B$v10452, B$v10445, B$v10447, B$v10449, B$v10451, filtro="B$v10440==1")
AddValidSumas("s10473", B$v10473, B$v10466, B$v10468, B$v10470, B$v10472, filtro="B$v10461==1")
AddValidSumas("s10494", B$v10494, B$v10487, B$v10489, B$v10491, B$v10493, filtro="B$v10482==1")
AddValidSumas("s10515", B$v10515, B$v10508, B$v10510, B$v10512, B$v10514, filtro="B$v10503==1")
AddValidSumas("s10536", B$v10536, B$v10529, B$v10531, B$v10533, B$v10535, filtro="B$v10524==1")
AddValidSumas("s10394", B$CIEN, B$v10394, B$v10395, B$v10396, filtro="B$v10387==1")
AddValidSumas("s10415", B$CIEN, B$v10415, B$v10416, B$v10417, filtro="B$v10408==1")
AddValidSumas("s10436", B$CIEN, B$v10436, B$v10437, B$v10438, filtro="B$v10429==1")
AddValidSumas("s10457", B$CIEN, B$v10457, B$v10458, B$v10459, filtro="B$v10450==1")
AddValidSumas("s10478", B$CIEN, B$v10478, B$v10479, B$v10480, filtro="B$v10471==1")
AddValidSumas("s10499", B$CIEN, B$v10499, B$v10500, B$v10501, filtro="B$v10492==1")
AddValidSumas("s10520", B$CIEN, B$v10520, B$v10521, B$v10522, filtro="B$v10513==1")
AddValidSumas("s10541", B$CIEN, B$v10541, B$v10542, B$v10543, filtro="B$v10534==1")
```

E. Validación de sumas: Capítulo 10.3 Desechos peligrosos.

```{r}
AddValidSumas("s10557", B$v10557, B$v10550, B$v10552, B$v10554, B$v10556, filtro="B$v10545==1")
AddValidSumas("s10578", B$v10578, B$v10571, B$v10573, B$v10575, B$v10577, filtro="B$v10566==1")
AddValidSumas("s10599", B$v10599, B$v10592, B$v10594, B$v10596, B$v10598, filtro="B$v10587==1")
AddValidSumas("s10620", B$v10620, B$v10613, B$v10615, B$v10617, B$v10619, filtro="B$v10608==1")
AddValidSumas("s10641", B$v10641, B$v10634, B$v10636, B$v10638, B$v10640, filtro="B$v10629==1")
AddValidSumas("s10662", B$v10662, B$v10655, B$v10657, B$v10659, B$v10661, filtro="B$v10650==1")
AddValidSumas("s10683", B$v10683, B$v10676, B$v10678, B$v10680, B$v10682, filtro="B$v10671==1")
AddValidSumas("s10704", B$v10704, B$v10697, B$v10699, B$v10701, B$v10703, filtro="B$v10692==1")
AddValidSumas("s10725", B$v10725, B$v10718, B$v10720, B$v10722, B$v10724, filtro="B$v10713==1")
AddValidSumas("s10746", B$v10746, B$v10739, B$v10741, B$v10743, B$v10745, filtro="B$v10734==1")
AddValidSumas("s10767", B$v10767, B$v10760, B$v10762, B$v10764, B$v10766, filtro="B$v10755==1")
AddValidSumas("s10788", B$v10788, B$v10781, B$v10783, B$v10785, B$v10787, filtro="B$v10776==1")
AddValidSumas("s10809", B$v10809, B$v10802, B$v10804, B$v10806, B$v10808, filtro="B$v10797==1")
AddValidSumas("s10830", B$v10830, B$v10823, B$v10825, B$v10827, B$v10829, filtro="B$v10818==1")
AddValidSumas("s10851", B$v10851, B$v10844, B$v10846, B$v10848, B$v10850, filtro="B$v10839==1")
AddValidSumas("s10872", B$v10872, B$v10865, B$v10867, B$v10869, B$v10871, filtro="B$v10860==1")
AddValidSumas("s10893", B$v10893, B$v10886, B$v10888, B$v10890, B$v10892, filtro="B$v10881==1")
AddValidSumas("s10914", B$v10914, B$v10907, B$v10909, B$v10911, B$v10913, filtro="B$v10902==1")
AddValidSumas("s10935", B$v10935, B$v10928, B$v10930, B$v10932, B$v10934, filtro="B$v10923==1")
AddValidSumas("s10956", B$v10956, B$v10949, B$v10951, B$v10953, B$v10955, filtro="B$v10944==1")
AddValidSumas("s10977", B$v10977, B$v10970, B$v10972, B$v10974, B$v10976, filtro="B$v10965==1")
AddValidSumas("s10998", B$v10998, B$v10991, B$v10993, B$v10995, B$v10997, filtro="B$v10986==1")
AddValidSumas("s10562", B$CIEN, B$v10562, B$v10563, B$v10564, filtro="B$v10555==1")
AddValidSumas("s10583", B$CIEN, B$v10583, B$v10584, B$v10585, filtro="B$v10576==1")
```

```
AddValidSumas("s10604", B$CIEN, B$v10604, B$v10605, B$v10606, filtro="B$v10597==1")
AddValidSumas("s10625", B$CIEN, B$v10625, B$v10626, B$v10627, filtro="B$v10618==1")
AddValidSumas("s10646", B$CIEN, B$v10646, B$v10647, B$v10648, filtro="B$v10639==1")
AddValidSumas("s10667", B$CIEN, B$v10667, B$v10668, B$v10669, filtro="B$v10660==1")
AddValidSumas("s10688", B$CIEN, B$v10688, B$v10689, B$v10690, filtro="B$v10681==1")
AddValidSumas("s10709", B$CIEN, B$v10709, B$v10710, B$v10711, filtro="B$v10702==1")
AddValidSumas("s10730", B$CIEN, B$v10730, B$v10731, B$v10732, filtro="B$v10723==1")
AddValidSumas("s10751", B$CIEN, B$v10751, B$v10752, B$v10753, filtro="B$v10744==1")
AddValidSumas("s10772", B$CIEN, B$v10772, B$v10773, B$v10774, filtro="B$v10765==1")
AddValidSumas("s10793", B$CIEN, B$v10793, B$v10794, B$v10795, filtro="B$v10786==1")
AddValidSumas("s10814", B$CIEN, B$v10814, B$v10815, B$v10816, filtro="B$v10807==1")
AddValidSumas("s10835", B$CIEN, B$v10835, B$v10836, B$v10837, filtro="B$v10828==1")
AddValidSumas("s10856", B$CIEN, B$v10856, B$v10857, B$v10858, filtro="B$v10849==1")
AddValidSumas("s10877", B$CIEN, B$v10877, B$v10878, B$v10879, filtro="B$v10870==1")
AddValidSumas("s10898", B$CIEN, B$v10898, B$v10899, B$v10900, filtro="B$v10891==1")
AddValidSumas("s10919", B$CIEN, B$v10919, B$v10920, B$v10921, filtro="B$v10912==1")
AddValidSumas("s10940", B$CIEN, B$v10940, B$v10941, B$v10942, filtro="B$v10933==1")
AddValidSumas("s10961", B$CIEN, B$v10961, B$v10962, B$v10963, filtro="B$v10954==1")
AddValidSumas("s10982", B$CIEN, B$v10982, B$v10983, B$v10984, filtro="B$v10975==1")
AddValidSumas("s11003", B$CIEN, B$v11003, B$v11004, B$v11005, filtro="B$v10996==1")
```

F. Validaciones lógicas: Capítulo 7.

```{r}
# Número de personas ocupadas desplegadas en la casilla 7001 debe coincidir con el valor de la casilla 5090.
AddRule("c7001", "B$v7001==B$v5090")

# Si selecciona la opción 5 "Ninguno", verificar que las opciones de la 1 a 4 no hayan sido seleccionadas.
AddRule("c71a", "B$v711==2 & B$v712==2 & B$v713==2 & B$v714==2", "B$v715==1")

# Si seleccionó al menos una de las opciones de la 1 a la 4, verificar que la opción 5 no haya sido seleccionada.
AddRule("c71b", "B$v715==2", "B$v711==1 | B$v712==1 | B$v713==1 | B$v714==1")

# Si seleccionó las opciones 1 y 2 al mismo tiempo, deberá especificar en el campo de Observaciones generales la razón de la existencia de las dos estructuras dentro de la empresa.
AddRule("c71c", "nchar(trimws(as.character(B$observaciones))) > 2", "B$v711==1 & B$v712==1")

# Obligatorio si respondió opciones 1, 2, 3 en la Pregunta 1. Validar si este valor es mayor que 0 y menor o igual que VAR_7001. En caso contrario, mostrar mensaje de error.
AddRule("c7002a", "B$v7002 > 0 & B$v7002 <= B$v7001", "B$v711==1 | B$v712==1 | B$v713==1")

# Obligatorio si respondió opciones 1, 2, 3 en la Pregunta 1. Validar si este valor es menor que el 50% del valor de la VAR_7001. En caso contrario, mostrar mensaje de error.
AddRule("c7002b", "B$v7002 > 0 & B$v7002 <= 0.5*B$v7001", "B$v711==1 | B$v712==1 | B$v713==1")

# Obligatorio si respondió a la Pregunta 7002. Validar si v7003 es mayor o igual que 0 y menor o igual que valor Pregunta 7002. En caso contrario, mostrar mensaje de error.
AddRule("c7003", "B$v7003 >= 0 & B$v7003 <= B$v7002", "B$v7002 > 0")

# Obligatorio si respondió a la Pregunta 7002. Validar si v7004 es mayor o igual que 0 y menor o igual que valor Pregunta 7002. En caso contrario, mostrar mensaje de error.
AddRule("c7004", "B$v7004 >= 0 & B$v7004 <= B$v7002", "B$v7002 > 0")

# La suma de v7003 y v7004 debe ser = al valor numérico ingresado en la Pregunta 7002.
AddValidSumas("c7002c", B$v7002, B$v7003, B$v7004, filtro="B$v7002 > 0")

# El valor numérico de la Pregunta 7005 debe ser mayor o igual que el sueldo básico por el personal (400*12* Valor(Pregunta 7003)) y no puede ser mayor que la variable 5180. Si el valor es menor al sueldo básico (400*12* Valor(v7003), mostrar advertencia y desplegar el campo de observaciones. No pasar sin observación.
AddRule("c7005", "B$v7005 >= 400*12*B$v7003 & B$v7005 <= B$v5180", "B$v7002 > 0")

# Si el valor de la suma de las variables v7005+v7006 es mayor que el registrado en la celda 5180, mostrar mensaje de error.
z <- apply(cbind(B$v7005, B$v7006), 1, sum, na.rm = T)
AddRule("c7006", "B$v5180 >= z", "B$v7002 > 0")
rm(z)
```

```
# Obligatorio si respondió opciones 1, 2, 3 y 4 en la Pregunta 1. Validar si v7007 es mayor o igual que 0 y menor o igual que VAR_1040. En caso contrario, mostrar mensaje de advertencia. (SOLO PARA MANUFACTURERAS).
filtrado <- "(B$v711 == 1 | B$v712 == 1 | B$v713 == 1 | B$v714 == 1) & substr(B$inec_ciiu4,1,1)=='C'"
AddRule("c7007", "B$v7007 >= 0 & B$v7007 <= B$v1040", filtrado)

# Obligatorio si respondió opciones 1, 2, 3 y 4 en la Pregunta 1. Validar si v7008 es mayor o igual que 0 y menor o igual que VAR_1041. En caso contrario, mostrar mensaje de advertencia.
AddRule("c7008", "B$v7008 >= 0 & B$v7008 <= B$v1041", filtrado)
rm(filtrado)

# Si en ambas variables v7008 y v7009 se registra el valor de 0, desplegar mensaje de advertencia y llenar la variable v7009 (Observaciones), no pasa si el campo de observaciones está vacío.
AddRule("c7009", "nchar(trimws(as.character(B$v7009))) > 2", "B$v7007==0 & B$v7008==0")

# Obligatorio. En el caso de contestar "Sí", continuar con la Pregunta 4.1; en el caso de contestar "No", pasar a la pregunta 5.
AddRule("c74", "B$v74 %in% 1:2")

# Pregunta activa únicamente si en la Pregunta 4 se respondió "SI". Validar que este valor sea mayor que cero. En caso contrario, mostrar mensaje de advertencia.
AddRule("c741", "B$v741 > 0", "B$v74==1")

# Pregunta activa únicamente si en la Pregunta 4.1 tiene valor mayor que cero. Validar que el valor ingresado en esta variable sea menor o igual al valor ingresado en la variable 4.1
AddRule("c742", "B$v742 >= 0 & B$v742 <= B$v741", "B$v741 > 0")

# Activa únicamente si se ha marcado como respuesta "otro", no pasa si la celda v7025 esta vacía.
AddRule("c7025", "nchar(trimws(as.character(B$v7025))) > 2", "B$v7024 > 0")

# En la variable 7030 debe desplegarse la suma automática de las variables 7014,7019,7024.
AddValidSumas("c7030a", B$v7030, B$v7014, B$v7019, B$v7024, filtro = "B$v742 > 0")

# En la variable 7030 debe desplegarse la suma automática de las variables 7026,7027,7028,7029.
AddValidSumas("c7030b", B$v7030, B$v7026, B$v7027, B$v7028, B$v7029, filtro = "B$v742 > 0")

# La variable 7030 debe ser igual a la variable 742, cuando esta última sea positiva.
AddRule("c7030c", "B$v7030==B$v742", "B$v742 > 0")

# Obligatorio. En el caso de contestar "Sí", continuar con la Pregunta 5.1; en el caso de contestar "No", pasar al Capítulo 8.
AddRule("c75", "B$v75 %in% 1:2")

# Obligatorio si respondió "Sí" en la Pregunta 5. Debe existir una sola respuesta (1 ó 2 ó 3 ó 4). En el caso de elegir "4. Otro", pasar a la pregunta 5.1.1. En los demás casos, avanzar al Capítulo 8.
AddRule("c751", "B$v751 %in% 1:4", "B$v75==1")

# Pregunta activa únicamente si en la Pregunta 5.1 se respondió "4. Otro".
AddRule("c7511", "nchar(trimws(as.character(B$v7511))) > 2", "B$v751==4")
```

G. Validaciones lógicas: Capítulo 8.

```{r}
# Obligatorio recuperar y desplegar información bajo la condición: v8001 = v4146 + v4147 + v4148 + v4196 + v4197.
AddValidSumas("c8001", B$v8001, B$v4146, B$v4147, B$v4148, B$v4196, B$v4197)
```

```{r}
# En las columnas 1,3 y 5 debe existir obligatoriamente una respuesta (1=SI o 2=NO).
seq_i <- seq(8002,8096,2)
filtrado <- "(B$v8098 > 0 | B$v8099 > 0 | B$v8100 > 0)"
for (i in seq_i) {
  etiqueta <- paste0("c", i)
  variable <- paste0("B$v", i)
  regla <- paste0(variable, " %in% 1:2")
  AddRule(etiqueta, regla, filtrado)
}
```

```{r}
# En el caso de que conteste que "SI" en las columnas 1,3,5, se debe registrar un valor > 0 en las columnas 2,4,6.
seq_i <- seq(8003,8097,2)
for (i in seq_i) {
  etiqueta <- paste0("c", i)
  variable <- paste0("B$v", i)
  previa <- paste0("B$v", i-1)
  regla <- paste0(variable, " > 0")
  filtrado <- paste0(previa, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(etiqueta,filtrado,i,previa,regla,seq_i,variable)
```

```{r}
# El total de la columna 2 (línea 328, VAR_8098) no puede ser superior que VAR_2006.
AddRule("c8098", "B$v8098 <= B$v2006")
```

```{r}
# El total de la columna 4 (línea 328, VAR_8099) no puede ser superior a la suma: v4146 + v4147 + v4148 + v4196 + v4197.
z <- apply(cbind(B$v4146, B$v4147, B$v4148, B$v4196, B$v4197), 1, sum, na.rm = T)
AddRule("c8099a", "B$v8099 <= z")
rm(z)
```

```{r}
# El total de la columna 4 (línea 328, VAR_8099) debe ser menor o igual al valor registrado en la variable v7007 del capítulo 7. Gestión ambiental.
AddRule("c8099b", "B$v8099 <= B$v7007")

# El total de la columna 6 (línea 328, VAR_8100) debe ser menor o igual a la v7008.
AddRule("c8100", "B$v8100 <= B$v7008")

# La variable 8053 debe ser <= que la variable 7007.
AddRule("c8053a", "B$v8053 <= B$v7007")

# La variable 8055 debe ser <= que la variable 7008.
AddRule("c8055a", "B$v8055 <= B$v7008")
```

H. Validaciones lógicas: Capítulo 9, Sección I.

```{r}
# En la pregunta 1, llenar obligatoriamente los campos de cantidad (2) y valor (3).
cond_v9001 <- "(B$v9001 == 0 | is.na(B$v9001))"
cond_v9002 <- "(B$v9002 == 0 | is.na(B$v9002))"
AddRule("c9003", "nchar(trimws(as.character(B$v9003))) > 2", paste0(cond_v9001, " | ", cond_v9002))
rm(cond_v9001, cond_v9002)

#  En la columna (3) se aplica la siguiente condición: VAR_9002 < VAR_1173.
AddRule("c9002", "B$v9002 < B$v1173", "B$v9002 > 0 & B$v1173 > 0")

# El Valor USD(3) dividido para la Cantidad(2) debe estar dentro del rango de 0.05 a 0.40.
AddRule("c9r01", "B$v9002 >= 0.05*B$v9001 & B$v9002 <= 0.4*B$v9001", "B$v9001 > 0")

# En la pregunta 2 debe existir una sola respuesta : "Sí" o "No". Si contesta "No", pasar a la Sección II. COMBUSTIBLES Y LUBRICANTES.
AddRule("c9i2", "B$v9i2 %in% 1:2")

# Si la variable v9044 está marcada con "SI", el aplicativo deberá solicitar información de forma obligatoria en la variable v9057.
AddRule("c9057", "nchar(trimws(as.character(B$v9057))) > 2", "B$v9044==1")

# La energía total producida (Var_9052) debe ser igual a la suma de la energía consumida (v9054) + la energía vendida (v9055).
AddValidSumas("c9052", B$v9052, B$v9054, B$v9055, filtro = "B$v9i2==1")

# En la columna 2 debe existir al menos un "Sí" (VAR_9004, VAR_9012, VAR_9020, VAR_9028, VAR_9036 o VAR_9044).
AddRule("c9i2a", "B$v9004==1 | B$v9012==1 | B$v9020==1 | B$v9028==1 | B$v9036==1 | B$v9044==1", "B$v9i2==1")
```

```
# Si al menos una de las variables: v9004,v9012,v9020,v9028,v9036,v9044 es igual a 1, entonces la Pregunta 2 debe responder que "SI".
AddRule("c9i2b", "B$v9i2==1", "B$v9004==1 | B$v9012==1 | B$v9020==1 | B$v9028==1 | B$v9036==1 | B$v9044==1")
```

```{r}
# Si responde "No" en la columna (2), bloquear la línea y pasar al siguiente tipo de energía.
# z <- apply(cbind(B$v9004, B$v9012, B$v9020, B$v9028, B$v9036, B$v9044), 1, sum, na.rm = T)
# condicion <- (z %in% 1:11)
# seq_i <- seq(9004,9044,8)
# for (i in seq_i) {
#   etiqueta <- paste0("c", i)
#   variable <- paste0("B$v", i)
#   secuencia <- NULL
#   for (j in 1:6) {
#     secuencia <- c(secuencia, paste0("v", i + j))
#   }
#   All_NA_0 <- apply(B[, secuencia], 1, function(x) {all(is.na(x) | x == 0)})
#   regla <- paste0(variable, "==2 & All_NA_0==TRUE")
#   AddRule(etiqueta, regla, "condicion==TRUE & B$v9i2==1")
# }
# rm(All_NA_0,condicion,etiqueta,i,j,regla,seq_i,secuencia,variable,z)
```

```{r}
# Obligatorio si se registra información en la columna 7 debe existir información en la columna 8.
seq_i <- seq(9010,9050,8)
for (i in seq_i) {
  etiqueta <- paste0("c", i)
  variable <- paste0("B$v", i)
  anterior <- paste0("B$v", i-1)
  regla <- paste0(variable, " > 0")
  condicion <- paste0(anterior, " > 0")
  AddRule(etiqueta, regla, condicion)
}
rm(condicion,etiqueta,i,regla,seq_i,anterior,variable)
```

```{r}
# La suma de los valores de las columnas (5) y (7) debe ser igual al valor registrado en la columna (3).
seq_i <- seq(9005,9045,8)
for (i in seq_i) {
  etiqueta <- paste0("c", i)
  variable <- paste0("B$v", i)
  sum_1 <- paste0("B$v", i+2)
  sum_2 <- paste0("B$v", i+4)
  filtrado <- paste0("B$v", i-1, "==1")
  comando <- paste0("AddValidSumas(\"", etiqueta, "\", ", variable, ", ", sum_1,
                    ", ", sum_2, ", filtro=\"", filtrado, "\")")
  eval(parse(text = comando), envir = .GlobalEnv)
}
rm(comando, etiqueta, filtrado,i,seq_i,sum_1,sum_2,variable)
```

```{r}
# Si existe valor en la columna 8 este debe ser >= que el valor de la columna 4.
seq_i <- seq(9010,9050,8)
for (i in seq_i) {
  etiqueta <- paste0("d", i)
  variable <- paste0("B$v", i)
  anterior <- paste0("B$v", i - 4)
  siguiente <- paste0("B$v", i + 1)
  regla1 <- paste0("(", variable, " >= ", anterior, ")")
  regla2 <- paste0("(", variable, " < ", anterior, ") & (nchar(trimws(as.character(", siguiente, "))) > 2)")
  regla <- paste0(regla1, " | ", regla2)
  filtrado <- paste0("(B$v", i-6, " ==1) & (B$v", i-5, " > B$v", i-3, ")")
  AddRule(etiqueta, regla, filtrado)
```

```
}
rm(anterior,etiqueta,filtrado,i,seq_i,regla1,regla2,regla,siguiente,variable)
```

```{r}
# Si los valores de la columna (7) son < que los valores de la columna (3), entonces obligatoriamente debe existir valores en las columnas (5) y (6).
seq_i <- seq(9008,9048,8)
for (i in seq_i) {
  etiqueta <- paste0("c", i)
  variable <- paste0("B$v", i)
  anterior <- paste0("B$v", i-1)
  regla <- paste0(variable, " > 0 & ", anterior, " > 0")
  condicion <- paste0("B$v", i+1, " < ", "B$v", i-3)
  AddRule(etiqueta, regla, condicion)
}
rm(anterior,etiqueta,condicion,i,seq_i,regla,variable)
```

```{r}
# En la pregunta 3 validar los valores de la columna VALOR (4) con respecto a la columna KWH/año (3), según la regla: "El Valor USD(4) dividido para los KWH/año(3) debe estar dentro del rango de 0.05 a 1.00".
AddRule("c9r11", "B$v9006 >= 0.05*B$v9005 & B$v9006 <= B$v9005", "B$v9005 > 0")
AddRule("c9r12", "B$v9014 >= 0.05*B$v9013 & B$v9014 <= B$v9013", "B$v9013 > 0")
AddRule("c9r13", "B$v9022 >= 0.05*B$v9021 & B$v9022 <= B$v9021", "B$v9021 > 0")
AddRule("c9r14", "B$v9030 >= 0.05*B$v9029 & B$v9030 <= B$v9029", "B$v9029 > 0")
AddRule("c9r15", "B$v9038 >= 0.05*B$v9037 & B$v9038 <= B$v9037", "B$v9037 > 0")
AddRule("c9r16", "B$v9046 >= 0.05*B$v9045 & B$v9046 <= B$v9045", "B$v9045 > 0")
```

```{r}
# Si se registran valores en la columna 7, no podrán ser mayores que los valores de la columna 3.
AddRule("d9009", "B$v9009 <= B$v9005", "B$v9005 > 0")
AddRule("d9017", "B$v9017 <= B$v9013", "B$v9013 > 0")
AddRule("d9025", "B$v9025 <= B$v9021", "B$v9021 > 0")
AddRule("d9033", "B$v9033 <= B$v9029", "B$v9029 > 0")
AddRule("d9041", "B$v9041 <= B$v9037", "B$v9037 > 0")
AddRule("d9049", "B$v9049 <= B$v9045", "B$v9045 > 0")
```

```{r}
# Si V9010, V9018, V9026, V9034 > 0 entonces en la línea 321, v8057 > 0.
AddRule("d8057", "B$v8057 > 0", "B$v9010 > 0 | B$v9018 > 0 | B$v9026 > 0 | B$v9034 > 0")
```

```{r}
# Si se registra en la columna (6) como opción de respuesta 3="Otros Usos", de forma obligatoria en la columna Observación (9) debe existir información.
AddRule("d9011", "nchar(trimws(as.character(B$v9011))) > 2", "B$v9008==3")
AddRule("d9019", "nchar(trimws(as.character(B$v9019))) > 2", "B$v9016==3")
AddRule("d9027", "nchar(trimws(as.character(B$v9027))) > 2", "B$v9024==3")
AddRule("d9035", "nchar(trimws(as.character(B$v9035))) > 2", "B$v9032==3")
AddRule("d9043", "nchar(trimws(as.character(B$v9043))) > 2", "B$v9040==3")
AddRule("d9051", "nchar(trimws(as.character(B$v9051))) > 2", "B$v9048==3")
```

I. Validaciones lógicas: Capítulo 9, Sección II.

```{r}
# En la pregunta 1, FILA 339, solamente para el casillero (9100) puede darse la opción 5="Mantenimiento".
for (i in seq(9060, 9096, 4)) {
  etiqueta <- paste0("c", i)
  regla <- paste0("B$v", i, " != 5")
  filtrado <- paste0("B$v", i - 2, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,etiqueta,regla, filtrado)
```

```{r}
# En la Pregunta 1, la columna Uso Principal (4) es de llenado obligatorio, siempre y cuando exista información en las columnas Cantidad (2) y Valor (3); así como no debe permitir ingresar información en la columna Uso Principal (4) si no existe información en las columnas Cantidad (2) y Valor (3).
for (i in seq(9060, 9096, 4)) {
  etiqueta <- paste0("d", i)
  regla <- paste0("B$v", i, " %in% c(1:4,6)")
  filtrado <- paste0("B$v", i - 2, " > 0 & B$v", i - 1, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
AddRule("d9100", "B$v9100 %in% 1:6", "B$v9098 > 0 & B$v9099 > 0")
rm(i,etiqueta,regla, filtrado)
```

```{r}
# Verificar que V9105 (linea 341, columna 3) sea <= a la suma de las V1130 + V1146.
z <- apply(cbind(B$v1130, B$v1146), 1, sum, na.rm = T)
AddRule("c9105", "B$v9105 <= z", "B$v9ii1==1")
rm(z)
```

```{r}
# Si se registra en la columna (4) como opción de respuesta 6="Otros Usos", de forma obligatoria en la columna Observación (5) debe existir información.
for (i in seq(9060, 9100, 4)) {
  etiqueta <- paste0("e", i)
  regla <- paste0("nchar(trimws(as.character(B$v", i + 1, "))) > 2")
  filtrado <- paste0("B$v", i, "==6")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,etiqueta,regla,filtrado)
```

J. Validaciones lógicas: Capítulo 10, Sección I.

```{r}
# En la pregunta 1, llenar obligatoriamente los campos de cantidad (2) y valor (3).
cond_v10000 <- "(B$v10000 == 0 | is.na(B$v10000))"
cond_v10001 <- "(B$v10001 == 0 | is.na(B$v10001))"
filtrado <- paste0(cond_v10000, " | ", cond_v10001)
AddRule("c10002", "nchar(trimws(as.character(B$v10002))) > 2", filtrado)
rm(cond_v10000, cond_v10001, filtrado)

#  En la columna (3) se aplica la siguiente condición: VAR_10001 < VAR_1173.
AddRule("c10001", "B$v10001 < B$v1173", "B$v10001 > 0")

# El Valor USD(3) dividido para la Cantidad(2) debe estar dentro del rango de 0.20 a 3.50.
AddRule("c10r01", "B$v10001 >= 0.20*B$v10000 & B$v10001 <= 3.50*B$v10000", "B$v10000 > 0")

# En la Pregunta 2, en el caso de contestar "SI" se sigue a la pregunta 2.1.
# En el caso de contestar "NO", se pasa a la Pregunta 3.
AddRule("c10i2", "B$v10i2 %in% 1:2")

# En la pregunta 2.1 Debe existir una sola respuesta numérica en la columna (1), (2) y (3). Esto para los que contestaron "SI" en la pregunta 2.
AddRule("c10i21", "(B$v10003 %in% 1:2) & (B$v10004 > 0) & (B$v10005 > 0)", "B$v10i2==1")

# En la Pregunta 3, en el caso de contestar "SI" sigue el flujo. En el caso de contestar "NO", pasar a la Sección II. Aguas Residuales/desechadas.
AddRule("c10i3", "B$v10i3 %in% 1:2")

# Si en la pregunta 3 se respondió que SI, debe haber por lo menos un "SI" por respuesta en la columna (1).
AddRule("c10i3a", "B$v10006==1 | B$v10014==1 | B$v10022==1", "B$v10i3==1")
```

```{r}
# En el caso de que en la columna (1) conteste "SI", en la columna (2) puede contestar cualquiera de las dos opciones.
AddRule("c10007", "B$v10007 %in% 1:2", "B$v10006==1")
```

```
AddRule("c10015", "B$v10015 %in% 1:2", "B$v10014==1")
AddRule("c10023", "B$v10023 %in% 1:2", "B$v10022==1")
```

```{r}
# Si en la columna (3) contesta "SI" debe haber valor en las columnas (4), (5) y (6).
AddRule("c10011", "B$v10009 > 0 & B$v10010 > 0 & B$v10011 > 0", "B$v10008==1")
AddRule("c10019", "B$v10017 > 0 & B$v10018 > 0 & B$v10019 > 0", "B$v10016==1")
AddRule("c10027", "B$v10025 > 0 & B$v10026 > 0 & B$v10027 > 0", "B$v10024==1")
```

```{r}
# Si en la columna (1) contesta "NO", automáticamente se deberá pasar a la siguiente fuente de captación de agua.
# for (i in seq(0,16,8)) {
#   secuencia <- NULL
#   for (j in 10007:10013) {
#     secuencia <- c(secuencia, paste0("v", i + j))
#   }
#   All_NA_0 <- apply(B[, secuencia], 1, function(x) {all(is.na(x) | x == 0)})
#   etiqueta <- paste0("aL", 342 + i/8)
#   filtrado <- paste0("B$v", 10006 + i, "==2")
#   AddRule(etiqueta, "isTRUE(All_NA_0)", filtrado)
# }
# rm(i,j,secuencia,All_NA_0,etiqueta,filtrado)
```

```{r}
# Si en la columna (3) contesta "NO", debe bloquear el ingreso de valores en las columnas (4),(5),(6) y (7).
# for (i in seq(0,16,8)) {
#   secuencia <- NULL
#   for (j in 10009:10012) {
#     secuencia <- c(secuencia, paste0("v", i + j))
#   }
#   All_NA_0 <- apply(B[, secuencia], 1, function(x) {all(is.na(x) | x == 0)})
#   etiqueta <- paste0("bL", 342 + i/8)
#   filtrado <- paste0("B$v", 10008 + i, "==2")
#   AddRule(etiqueta, "isTRUE(All_NA_0)", filtrado)
# }
# rm(i,j,secuencia,All_NA_0,etiqueta,filtrado)
```

```{r}
# La información registrada en las variables 10012,10020 y 10028 debe existir respuesta de formato numérico de escala (F10.2).
AddRule("c10012", "(100*B$v10012) %% 100 >= 0", "B$v10008==1")
AddRule("c10020", "(100*B$v10020) %% 100 >= 0", "B$v10016==1")
AddRule("c10028", "(100*B$v10028) %% 100 >= 0", "B$v10024==1")
```

```{r}
# Si en la columna (1) contesta que "SI", debe haber obligatoriamente valor en la columna (8).
AddRule("c10013", "B$v10013 > 0", "B$v10006==1")
AddRule("c10021", "B$v10021 > 0", "B$v10014==1")
AddRule("c10029", "B$v10029 > 0", "B$v10022==1")
```

```{r}
# Las variables 10010, 10018, 10026 deben ser menores o iguales que 24.
AddRule("d10010", "B$v10010 > 0 & B$v10010 <= 24", "B$v10008==1")
AddRule("d10018", "B$v10018 > 0 & B$v10018 <= 24", "B$v10016==1")
AddRule("d10026", "B$v10026 > 0 & B$v10026 <= 24", "B$v10024==1")
```

```{r}
# Las variables 10011, 10019, 10027 deben ser menores o iguales que 31.
AddRule("d10011", "B$v10011 > 0 & B$v10011 <= 31", "B$v10008==1")
AddRule("d10019", "B$v10019 > 0 & B$v10019 <= 31", "B$v10016==1")
AddRule("d10027", "B$v10027 > 0 & B$v10027 <= 31", "B$v10024==1")
```

```
```

```{r}
# La información registrada en las variables 10009, 10017 y 10025 debe existir respuesta de formato numérico de escala (F10.2).
AddRule("c10009", "(100*B$v10009) %% 100 >= 0", "B$v10008==1")
AddRule("c10017", "(100*B$v10017) %% 100 >= 0", "B$v10016==1")
AddRule("c10025", "(100*B$v10025) %% 100 >= 0", "B$v10024==1")
```

```{r}
# En el casillero v10012 se despliega el cálculo de la fórmula: v10012 = 12 * v10009 * v10010 * v10011.
# G$DOCE <- 12
# z <- apply(cbind(G$DOCE, B$v10009, B$v10010, B$v10011), 1, prod, na.rm = F)
# AddRule("d10012", "B$v10012 == z", "B$v10008==1")
# rm(z)
```

```{r}
# En el casillero v10020 se despliega el cálculo de la fórmula: v10020 = 12 * v10017 * v10018 * v10019.
# z <- apply(cbind(G$DOCE, B$v10017, B$v10018, B$v10019), 1, prod, na.rm = F)
# AddRule("d10020", "B$v10020 == z", "B$v10016==1")
# rm(z)
```

```{r}
# En el casillero v10020 se despliega el cálculo de la fórmula: v10028 = 12 * v10025 * v10026 * v10027.
# z <- apply(cbind(G$DOCE, B$v10025, B$v10026, B$v10027), 1, prod, na.rm = F)
# AddRule("d10028", "B$v10028 == z", "B$v10024==1")
# rm(z)
# G$DOCE <- NULL
```

```{r}
# Verificar la sumatoria de las variables: v9002 + v10001 < v1173.
z <- apply(cbind(B$v9002, B$v10001), 1, sum, na.rm = T)
AddRule("c1173", "z < B$v1173")
rm(z)
```

K. Validaciones lógicas: Capítulo 10, Sección II.

```{r}
# Obligatorio. En el caso de contestar "Sí", ir a la pregunta 1.1; en el caso de contestar "No", pasar a la pregunta 2.
AddRule("c10ii1", "B$v10ii1 %in% 1:2")

# Obligatorio si respondió "Sí" en la Pregunta 1. Esta variable debe admitir como valor mínimo 1.0.
AddRule("c10ii11", "B$v10ii11 >= 1", "B$v10ii1==1")

# Obligatorio. Debe existir una sola respuesta "Sí" ó "No". Si el informante contesta "No", pasar directamente a III. Otros
Residuos y/o desechos.
AddRule("c10ii2", "B$v10ii2 %in% 1:2")

# Obligatorio. Debe existir una sola respuesta "Sí" ó "No". Si el informante contesta "No", pasar directamente a la pregunta 5.
AddRule("c10ii3", "B$v10ii3 %in% 1:2", "B$v10ii2==1")
```

```{r}
# Debe existir respuesta de formato numérico de escala (F10.2) en las columnas (1), (2), (3) y (4).
oper <- function(x) {((100*x) %% 100) >= 0}
regla1 <- "oper(B$v10032) & oper(B$v10033) & oper(B$v10034) & oper(B$v10035)"
AddRule("c10ii4", regla1, "B$v10ii3==1")
rm(oper,regla1)
```
```{r}
# En el casillero v10035 se despliega el cálculo de la fórmula: v10035 = 12 * v10032 * v10033 * v10034.
# B$DOCE <- 12
# z <- apply(cbind(B$DOCE, B$v10032, B$v10033, B$v10034), 1, prod, na.rm = F)
# AddRule("c10035", "B$v10035 == z", "B$v10008==1")
```

```
# rm(z)
# B$DOCE <- NULL
```

```{r}
# El valor de la v10035 debe ser inferior o igual que la suma de las variables del Cap. 10, Sección I (Agua): v10000 (Preg. 1) + v10004 (Preg. 3.1) + v10030 (Preg. 4).
Agua_Tank <- ifelse(B$v10003 == 1, B$v10004, ifelse(B$v10003 == 2, B$v10004/264.172, NA))
Agua <- apply(cbind(B$v10000, Agua_Tank, B$v10030), 1, sum, na.rm = T)
AddRule("d10035", "B$v10035 <= Agua", "B$v10ii3==1")
rm(Agua, Agua_Tank)
```

```{r}
# Si en la pregunta 3 contesto "SI", obligatoriamente debe existir información en las columnas (1), (2) y (3).
AddRule("c10034", "B$v10032 > 0 & B$v10033 > 0 & B$v10034 > 0", "B$v10ii3==1")
```

```{r}
# Las variables: v10033 <= 24 y v$10034 <= 31.
AddRule("c10033", "B$v10033 <= 24 & B$v10034 <= 31", "B$v10ii3==1")
```

```{r}
# Obligatorio. En el caso de contestar "Sí", continuar con la Pregunta 5.1; en el caso de contestar "No", pasar al Capítulo 8.
AddRule("c10ii5", "B$v10ii5 %in% 1:2", "B$v10ii2==1")
AddRule("c10ii5111", "B$v10ii5111 %in% 1:2", "B$v10ii2==1 & B$v10ii5==1")
AddRule("c10ii5112", "B$v10ii5112 %in% 1:2", "B$v10ii2==1 & B$v10ii5==1")
AddRule("c10ii5113", "B$v10ii5113 %in% 1:2", "B$v10ii2==1 & B$v10ii5==1")
AddRule("c10ii5114", "B$v10ii5114 %in% 1:2", "B$v10ii2==1 & B$v10ii5==1")
```

```{r}
#  Si está marcada la variable 2"Ninguno" con "SI", entonces las variables 1.1,1.2,1.3,1.4, no deberán estar marcadas.
regla <- NULL
regla[1] <- "B$v10ii5111 %in% c(0,NA)"
regla[2] <- "B$v10ii5112 %in% c(0,NA)"
regla[3] <- "B$v10ii5113 %in% c(0,NA)"
regla[4] <- "B$v10ii5114 %in% c(0,NA)"
regla <- paste0("(", regla[1], " & ", regla[2], " & ", regla[3], " & ", regla[4], ")")
AddRule("e10ii5", regla, "B$v10ii2==1 & B$v10ii5==2")
rm(regla)
```

```{r}
# Si en el Capítulo 10, Sección II, Pregunta 5, se escoge la opción "SI" en 1. Procesos, entonces la Línea 313 del Capítulo 8 deberá tener valores > 0.
r1 <- "B$v10ii5==1"
r2 <- "B$v10ii5111==1"
r3 <- "B$v10ii5112==1"
r4 <- "B$v10ii5113==1"
regla1 <- paste0(r1, " | ", r2, " | ", r3, " | ", r4)
r6 <- "B$v8009 > 0"
r7 <- "B$v8011 > 0"
r8 <- "B$v8013 > 0"
regla2 <- paste0(r6, " | ", r7, " | ", r8)
regla3 <- paste0("(", regla1, ") & (", regla2, ")")
AddRule("cL313", regla3, "B$v10ii2==1")
rm(r1,r2,r3,r4,r6,r7,r8,regla1,regla2,regla3)
```

```{r}
# Si en la Pregunta 5 se contestó a 2.- Ninguno = "SI", el valor de la Pregunta 6 debe ser 0%.
AddRule("d10ii6", "B$v10ii6==0", "B$v10ii5==2")
```

```{r}
# En la Pregunta 6 debe existir una respuesta numérica entre 0% y 100%.
AddRule("c10ii6", "B$v10ii6 %in% 0:100", "B$v10ii2==1 & B$v10ii5==1")
```

```{r}
# Si el valor registrado en la Pregunta 6 está entre 1% y 99%, debe existir información en las preguntas 7 y 8.
regla <- ""
seq_7 <- seq(10037,10047,2)
for (i in seq_7) {
  sec_7 <- paste0("B$v", i)
  pre_7 <- paste0("B$v", i - 1)
  sec_8 <- paste0("B$v", i + 13)
  pre_8 <- paste0("B$v", i + 12)
  temp_71 <- paste0("(", pre_7, " == 1 & ", sec_7, " %in% 0:100)")
  temp_72 <- paste0("(", pre_7, " == 2 & is.na(", sec_7, "))")
  temp_81 <- paste0("(", pre_8, " == 1 & ", sec_8, " %in% 0:100)")
  temp_82 <- paste0("(", pre_8, " == 2 & is.na(", sec_8, "))")
  temp <- paste0("(", temp_71, " | ", temp_72, ") & (", temp_81, " | ", temp_82, ")")
  regla <- ifelse(regla == "", temp, paste0("(", regla, " | ", temp, ")"))
}
regla <- paste0(regla, " & B$v10048 == 100 & B$v10061 == 100")
AddRule("e10ii6", regla, "B$v10ii2==1 & B$v10ii5==1 & (B$v10ii6 %in% 1:99)")
rm(i,seq_7,regla,sec_7,pre_7,sec_8,pre_8,temp_71,temp_72,temp_81,temp_82,temp)
```

```{r}
# Si la Pregunta 6 vale 0%, pasar a la Pregunta 8 con bloqueo de la Pregunta 7.
regla <- ""
seq_7 <- seq(10037,10047,2)
for (i in seq_7) {
  sec_7 <- paste0("B$v", i)
  pre_7 <- paste0("B$v", i - 1)
  sec_8 <- paste0("B$v", i + 13)
  pre_8 <- paste0("B$v", i + 12)
  temp_71 <- paste0("is.na(", pre_7, ") & is.na(", sec_7, ")")
  temp_72 <- paste0("is.na(", pre_7, ") & is.na(", sec_7, ")")
  temp_81 <- paste0("(", pre_8, " == 1 & ", sec_8, " %in% 0:100)")
  temp_82 <- paste0("(", pre_8, " == 2 & is.na(", sec_8, "))")
  temp <- paste0("(", temp_71, " | ", temp_72, ") & (", temp_81, " | ", temp_82, ")")
  regla <- ifelse(regla == "", temp, paste0("(", regla, " | ", temp, ")"))
}
regla <- paste0(regla, " & is.na(B$v10048) & B$v10061 == 100")
AddRule("f10ii6", regla, "B$v10ii2==1 & B$v10ii5==2 & (B$v10ii6==0)")
rm(i,seq_7,regla,sec_7,pre_7,sec_8,pre_8,temp_71,temp_72,temp_81,temp_82,temp)
```

```{r}
# Si la Pregunta 6 vale 100%, pasar a la Pregunta 7 con bloqueo de la Pregunta 8.
regla <- ""
seq_7 <- seq(10037,10047,2)
for (i in seq_7) {
  sec_7 <- paste0("B$v", i)
  pre_7 <- paste0("B$v", i - 1)
  sec_8 <- paste0("B$v", i + 13)
  pre_8 <- paste0("B$v", i + 12)
  temp_71 <- paste0("(", pre_7, " == 1 & ", sec_7, " %in% 0:100)")
  temp_72 <- paste0("(", pre_7, " == 2 & is.na(", sec_7, "))")
  temp_81 <- paste0("is.na(", pre_8, ") & is.na(", sec_8, ")")
  temp_82 <- paste0("is.na(", pre_8, ") & is.na(", sec_8, ")")
  temp <- paste0("(", temp_71, " | ", temp_72, ") & (", temp_81, " | ", temp_82, ")")
  regla <- ifelse(regla == "", temp, paste0("(", regla, " | ", temp, ")"))
}
regla <- paste0(regla, " & B$v10048 == 100 & is.na(B$v10061)")
AddRule("g10ii6", regla, "B$v10ii2==1 & B$v10ii5==1 & (B$v10ii6==100)")
rm(i,seq_7,regla,sec_7,pre_7,sec_8,pre_8,temp_71,temp_72,temp_81,temp_82,temp)
```

```{r}
# En el caso de contestar "Sí" en la columna (1) de la Pregunta 7, registrar los respectivos porcentajes.
AddRule("d10036", "B$v10037 %in% 1:100", "B$v10036==1")
AddRule("d10038", "B$v10039 %in% 1:100", "B$v10038==1")
AddRule("d10040", "B$v10041 %in% 1:100", "B$v10040==1")
AddRule("d10042", "B$v10043 %in% 1:100", "B$v10042==1")
AddRule("d10044", "B$v10045 %in% 1:100", "B$v10044==1")
AddRule("d10046", "B$v10047 %in% 1:100", "B$v10046==1")
```

```{r}
# En el caso de contestar "Sí" en la columna (1) de la Pregunta 8, registrar los respectivos porcentajes.
AddRule("d10049", "B$v10050 %in% 1:100", "B$v10049==1")
AddRule("d10051", "B$v10052 %in% 1:100", "B$v10051==1")
AddRule("d10053", "B$v10054 %in% 1:100", "B$v10053==1")
AddRule("d10055", "B$v10056 %in% 1:100", "B$v10055==1")
AddRule("d10057", "B$v10058 %in% 1:100", "B$v10057==1")
AddRule("d10059", "B$v10060 %in% 1:100", "B$v10059==1")
```

L. Validaciones lógicas: Capítulo 10, Sección III (Residuos y Desechos).

```{r}
# La información de la columna (1) es obligatoria (responder "Sí" ó "No").
seq_i <- seq(10062,10986,21)
for (i in seq_i) {
  etiqueta <- paste0("c", i)
  variable <- paste0("B$v", i)
  regla <- paste0(variable, " %in% 1:2")
  AddRule(etiqueta, regla)
}
rm(i,seq_i,etiqueta,variable,regla)
```

```{r}
# Si responde "Sí" en la columna (1) y "No" en la columna (1.1), debe contestar obligatoriamente en las columnas (3.3) y (4.1). En al menos una de ellas debe haber un "Sí". En caso contrario, mostrar mensaje de advertencia.

for (i in seq(0,924,21)) {
  regla_3_3 <- paste0("(B$v", 10070 + i, " %in% 1:2)")
  regla_4_1 <- paste0("(B$v", 10072 + i, " %in% 1:2)")

  vars_suma <- paste0('c(\"v', 10070 + i, '\", \"v', 10072 + i, '\")')
  suma_col <- apply(B[, eval(parse(text = vars_suma))], 1, sum, na.rm = T)
  regla_5 <- "suma_col %in% 2:3"

  etiqueta <- paste0("dL", 361 + i/21)

  regla <- paste0(regla_3_3, " & ", regla_4_1, " & ", regla_5)

  filtrado <- paste0("B$v", 10062 + i, "==1 & B$v", 10063 + i, "==2")

  AddRule(etiqueta, regla, filtrado)
}
rm(i,regla_3_3,regla_4_1,vars_suma,suma_col,regla_5,etiqueta,regla,filtrado)
```

```{r}
# Si responde "Sí" en la columna (1) y "Sí" en la columna (1.1), debe contestar obligatoriamente en las columnas (3.1.1), (3.2.1), (3.3.1), (4.1.1). En al menos una de ellas debe haber un "Sí". En caso contrario, mostrar mensaje de advertencia.

for (i in seq(0,924,21)) {
  regla_3_1_1 <- paste0("(B$v", 10066 + i, " %in% 1:2)")
  regla_3_2_1 <- paste0("(B$v", 10068 + i, " %in% 1:2)")
  regla_3_3_1 <- paste0("(B$v", 10070 + i, " %in% 1:2)")
  regla_4_1_1 <- paste0("(B$v", 10072 + i, " %in% 1:2)")

  vars_suma <- paste0('c(\"v', 10066 + i, '\", "v', 10068 + i, '\", "v', 10070 + i, '\", \"v', 10072 + i, '\")')
```

```
  suma_col <- apply(B[, eval(parse(text = vars_suma))], 1, sum, na.rm = T)
  regla_5 <- "suma_col %in% 4:7"

  etiqueta <- paste0("tL", 361 + i/21)

  regla <- paste0(regla_3_1_1, " & ", regla_3_2_1, " & ", regla_3_3_1, " & ", regla_4_1_1, " & ", regla_5)

  filtrado <- paste0("B$v", 10062 + i, "==1 & B$v", 10063 + i, "==1")

  AddRule(etiqueta, regla, filtrado)
}
rm(i,regla_3_1_1,regla_3_2_1,regla_3_3_1,regla_4_1_1,vars_suma,suma_col,regla_5, etiqueta, regla, filtrado)
```

```{r}
# Si contesta "NO" en la columna 1.1 debe bloquearse las columnas 2.1, 2.2, 3.1 y 3.2. Para este caso, se debe registrar valores
en las columnas 3.3 y 4.1.

for (i in seq(0,924,21)) {
  regla_2 <- paste0("((is.na(B$v", 10064 + i, ") | B$v", 10064 + i,
                    " == 0) & (is.na(B$v", 10065 + i, ") | B$v", 10065 + i, " == 0))")
  regla_3_1 <- paste0("((is.na(B$v", 10066 + i, ") | B$v", 10066 + i,
                    " == 0) & (is.na(B$v", 10067 + i, ") | B$v", 10067 + i, " == 0))")
  regla_3_2 <- paste0("((is.na(B$v", 10068 + i, ") | B$v", 10068 + i,
                    " == 0) & (is.na(B$v", 10069 + i, ") | B$v", 10069 + i, " == 0))")

  regla_3_3a <- paste0("(B$v", 10070 + i, " ==1 & B$v", 10071 + i, " > 0)")
  regla_3_3b <- paste0("(B$v", 10070 + i, " ==2 & (is.na(B$v", 10071 + i,
                     ") | B$v", 10071 + i, " == 0))")
  regla_3_3 <- paste0("(", regla_3_3a, " | ", regla_3_3b, ")")

  regla_4_1a <- paste0("(B$v", 10072 + i, " ==1 & B$v", 10073 + i, " > 0)")
  regla_4_1b <- paste0("(B$v", 10072 + i, " ==2 & (is.na(B$v", 10073 + i,
                     ") | B$v", 10073 + i, " == 0))")
  regla_4_1 <- paste0("(", regla_4_1a, " | ", regla_4_1b, ")")

  regla_x <- paste0("!(", regla_3_3b, " & ", regla_4_1b, ")")

  etiqueta <- paste0("fL", 361 + i/21)
  regla <- paste0("(", regla_2, " & ", regla_3_1, " & ", regla_3_2, ") & ((",
                  regla_3_3, " | ", regla_4_1, ") & ", regla_x, ")")
  filtrado <- paste0("B$v", 10063 + i, "==2")
  AddRule(etiqueta, regla, filtrado)
}
rm(i, regla_2, regla_3_1, regla_3_2, regla_3_3a, regla_3_3b, regla_3_3, regla_4_1a, regla_4_1b, regla_4_1, regla_x, etiqueta,
regla, filtrado)
```

```{r}
# En la columna (2.1) de la pregunta 1, debe existir las opciones de unidades de medida siguientes: 1 Kilogramo; 2 Tonelada.

for (i in seq(10064,10358,21)) {
  etiqueta <- paste0("c", i)
  regla <- paste0("B$v", i, " %in% 1:2")
  filtrado <- paste0("B$v", i - 2, " == 1 & B$v", i - 1, " == 1 & B$v", i + 1, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(etiqueta,regla,filtrado)
```

```{r}
# En la columna (2.1) de la pregunta 2, debe existir las opciones de unidades de medida siguientes: 1 kg, 2 ton en casi todas las
líneas, excepto en la línea 377, casillero v10400, donde se permitirá registrar: 1 Kilogramo; 2 Tonelada; 3 Galón.
for (i in seq(10379,10526,21)) {
  etiqueta <- paste0("c", i)
  regla1 <- paste0("B$v", i, " %in% 1:3")
  regla2 <- paste0("B$v", i, " %in% 1:2")
```

```
  regla <- ifelse(i == 10400, regla1, regla2)
  filtrado <- paste0("B$v", i - 2, " == 1 & B$v", i - 1, " == 1 & B$v", i + 1, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(etiqueta,regla,regla1, regla2,filtrado)
```

```{r}
# En la columna (2.1) de la pregunta 3, debe existir las opciones de unidades de medida siguientes: 1 Kilogramo; 2 Tonelada; 3 Galón.
for (i in seq(10547,10988,21)) {
  etiqueta <- paste0("c", i)
  regla <- paste0("B$v", i, " %in% 1:3")
  filtrado <- paste0("B$v", i - 2, " == 1 & B$v", i - 1, " == 1 & B$v", i + 1, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(etiqueta,regla,filtrado)
```

```{r}
# Debe existir una respuesta en cada columna (3.1.1) (3.2.1) (3.3.1) y (4.1.1).
for (i in seq(0,924,21)) {
  variable <- NULL
  regla <- NULL
  for (j in seq(10066,10072,2)) {
    variable <- paste0("B$v", i + j)
    regla <- ifelse(is.null(regla), paste0("(", variable, " %in% 1:2)"),
                    paste0(regla, " & (", variable, " %in% 1:2)"))
  }
  etiqueta <- paste0("gL", 361 + i/21)
  filtrado <- paste0("B$v", 10063 + i, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,j,variable,regla,etiqueta,filtrado)
```

```{r}
# Si responde "Sí" en alguna de las columnas (3.1.1), (3.2.1), (3.3.1) o (4.1.1), llenar con un valor numérico entero mayor que cero las variables compañeras (3.1.2), (3.2.2), (3.3.2) y (4.1.2), siempre y cuando la columna (1.1) tenga por respuesta "Sí".
for (i in seq(0,924,21)) {
  regla <- NULL
  for (j in seq(10066,10072,2)) {
    variable_A <- paste0("B$v", i + j)
    variable_B <- paste0("B$v", i + j + 1)
    n <- ifelse((j %% 8)/2 == 0, 4, (j %% 8)/2)
    regla[n] <- paste0("(", variable_A, " == 1) & (", variable_B, " > 0)")
  }
  regla <- paste0("(", regla[1], ") | (", regla[2], ") | (", regla[3], ") | (", regla[4], ")")
  etiqueta <- paste0("hL", 361 + i/21)
  filtrado <- paste0("B$v", 10063 + i, "==1 & B$v", 10065 + i, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,j,n,variable_A,variable_B,regla,etiqueta,filtrado)
```

```{r}
# Caso contrario, si en la columna (1.1) registra como respuesta "No", permítase en las columnas (3.3.2) y (4.1.2) ingresar valores iguales a cero únicamente.
for (i in seq(0,924,21)) {
  regla <- NULL
  for (j in seq(10070,10072,2)) {
    variable_A <- paste0("B$v", i + j)
    variable_B <- paste0("B$v", i + j + 1)
    subregla1 <- paste0("(", variable_A, " == 1) & (", variable_B, " == 0)")
    subregla2 <- paste0("(", variable_A, " == 2) & is.na(", variable_B, ")")
    n <- ifelse((j %% 4)/2 == 0, 2, 1)
    regla[n] <- paste0("(", subregla1, ") | (", subregla2, ")")
```

```
  }
  regla <- paste0("(", regla[1], ") & (", regla[2], ")")
  etiqueta <- paste0("iL", 361 + i/21)
  filtrado <- paste0("B$v", 10063 + i, "==2")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,j,n,variable_A,variable_B,subregla1,subregla2,regla,etiqueta,filtrado)
```

```{r}
# La suma (columna 5) debe ser igual a la cantidad registrada en la columna 2.2, siempre y cuando la columna (1.1) tenga por
respuesta "Sí".
for (i in seq(0,924,21)) {
  regla <- NULL
  variable_A <- paste0("B$v", i + 10074)
  variable_B <- paste0("B$v", i + 10065)
  regla <- paste0("(", variable_A, " == ", variable_B, ")")
  etiqueta <- paste0("jL", 361 + i/21)
  filtrado <- paste0("B$v", 10065 + i, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,variable_A,variable_B,regla,etiqueta,filtrado)
```

```{r}
# En la columna (2.2), si existe un valor mayor a 0 debe verificarse que exista información en al menos una de las columnas
(3.1.2), (3.2.2), (3.3.2), (4.1.2).
for (i in seq(0,924,21)) {
  regla <- NULL
  for (j in seq(10066,10072,2)) {
    variable_A <- paste0("B$v", i + j)
    variable_B <- paste0("B$v", i + j + 1)
    n <- ifelse((j %% 8)/2 == 0, 4, (j %% 8)/2)
    regla[n] <- paste0("(", variable_A, " == 1) & (", variable_B, " > 0)")
  }
  regla <- paste0("(", regla[1], ") | (", regla[2], ") | (", regla[3], ") | (", regla[4], ")")
  etiqueta <- paste0("kL", 361 + i/21)
  filtrado <- paste0("B$v", 10065 + i, " > 0")
  AddRule(etiqueta, regla, filtrado)
}
rm(i, j, n, variable_A, variable_B, regla, etiqueta, filtrado)
```

```{r}
# Si en la columna (4.1.1) se respondió "Sí", llenar desde la columna (6.1) hasta la columna (7.2) obligatoriamente.
for (i in seq(0,924,21)) {
  regla <- NULL
  variable <- NULL
  for (j in 1:4) {
    variable[j] <- paste0("B$v", 10074 + i + j)
  }
  regla[1] <- paste0("(", variable[1], "==1 & ", variable[2], " > 0)")
  regla[2] <- paste0("(", variable[1], "==2 & ", "is.na(", variable[2], "))")
  regla[3] <- paste0("(", variable[3], "==1 & ", variable[4], " > 0)")
  regla[4] <- paste0("(", variable[3], "==2 & ", "is.na(", variable[4], "))")
  regla[5] <- paste0("(", regla[1], " & ", regla[3], ")")
  regla[6] <- paste0("(", regla[1], " & ", regla[4], ")")
  regla[7] <- paste0("(", regla[2], " & ", regla[3], ")")
  regla[8] <- paste0("(", regla[2], " & ", regla[4], ")")
  regla <- paste0("(", regla[5], " | ", regla[6], " | ", regla[7], " | ", regla[8], ")")
  etiqueta <- paste0("mL", 361 + i/21)
  filtrado <- paste0("B$v", 10072 + i, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,j,variable,regla,etiqueta,filtrado)
```

```{r}
# Si en la columna (4.1.1) se respondió "Sí", en las columnas (8.1), (8.2) y (8.3) debe existir un porcentaje válido.
for (i in seq(0,924,21)) {
  regla <- NULL
  for (j in 0:2) {
    variable <- paste0("B$v", 10079 + i + j)
    regla[j + 1] <- paste0(variable, " %in% 0:100")
  }
  regla <- paste0("(", regla[1], ") | (", regla[2], ") | (", regla[3], ")")
  etiqueta <- paste0("nL", 361 + i/21)
  filtrado <- paste0("B$v", 10072 + i, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,j,variable,regla,etiqueta,filtrado)
```

```{r}
# Si en la columna (4.1.1) se respondió "Si", de forma obligatoria debe existir información en las columnas (2.1) y (2.2), siempre y cuando en la columna (8.2) exista un valor mayor que cero.
for (i in seq(0,924,21)) {
  regla <- NULL
  regla[1] <- paste0("(B$v", 10064 + i, " %in% 1:3 & B$v", 10065 + i, " > 0)")
  regla[2] <- paste0("(B$v", 10080 + i, " > 0)")
  regla <- paste0("!", regla[2], " | ", regla[1])
  etiqueta <- paste0("qL", 361 + i/21)
  filtrado <- paste0("B$v", 10072 + i, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,regla,etiqueta,filtrado)
```

```{r}
# Si en el Capítulo 10, Sección III (Otros residuos y/o desechos), Preguntas 1,2,3, Columna 6.2, existe algún valor mayor que cero, entonces VAR_8019 > 0.
for (i in seq(0,924,21)) {
  regla <- NULL
  regla <- paste0("!(B$v", 10076 + i, "> 0) | B$v8019 > 0")
  etiqueta <- paste0("c", 10076 + i)
  filtrado <- paste0("B$v", 10075 + i, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,regla,etiqueta,filtrado)
```

```{r}
# La Suma de todos los valores ingresados en la columna 6.2 no pueden ser mayor al casillero v8019.
for (i in seq(0,924,21)) {
  regla <- NULL
  regla <- paste0("B$v", 10076 + i, "<= B$v8019")
  etiqueta <- paste0("d", 10076 + i)
  filtrado <- paste0("B$v", 10075 + i, "==1")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,regla,etiqueta,filtrado)
```

```{r}
# La columna (9) (Observación: Especifique otro tipo de recolector) se llena cuando en la columna (8.3) (Otro) se ha respondido con un porcentaje mayor que 0%.
for (i in seq(0,924,21)) {
  regla <- NULL
  regla <- paste0("nchar(trimws(as.character(B$v", 10082 + i, "))) > 2")
  etiqueta <- paste0("c", 10082 + i)
  filtrado <- paste0("B$v", 10081 + i, " %in% 1:100")
  AddRule(etiqueta, regla, filtrado)
}
rm(i,regla,etiqueta,filtrado)
```

```
```

```{r}
# Obligatorio. En el caso de contestar "Sí", ir a la pregunta 4.1; en el caso de contestar "No", pasar al Capítulo siguiente.
AddRule("c10iii4", "B$v10iii4 %in% 1:2", "B$efectividad == 1")

# Obligatorio si respondió "Sí" en la Pregunta 4. En el caso de contestar "Sí", ir a la pregunta 4.2; en el caso de contestar "No", pasar al Capítulo siguiente.
AddRule("c10iii41", "B$v10iii41 %in% 1:2", "B$v10iii4 == 1")

# Obligatorio si respondió "SI" en la pregunta 4.1.
AddRule("c10iii42", "B$v10iii42 > 0", "B$v10iii41 == 1")

# El valor de la Var 4.2 debe ser menor que la variable Var_3181.
AddRule("d10iii42", "B$v10iii42 < B$v3181", "B$v10iii41 == 1")
```

```{r}
# Si marcó "SI" en la columna (1.1), pero no se ingresó valores en las columnas (2.1) y (2.2), entonces, en el caso de que en alguna de las columnas (3.1.1), (3.2.1), (3.3.1), (4.1.1) se haya seleccionado "Sí", verificar que en las columnas (3.1.2), (3.2.2), (3.3.2), (4.1.2) haya únicamente el valor cero.
for (i in seq(0,924,21)) {
  regla <- NULL
  for (j in seq(0,6,2)) {
    regla[j/2 + 1] <- paste0("(B$v", 10066 + i + j, " == 1 & B$v", 10067 + i + j, " == 0)")
  }
  regla <- paste0(regla[1]," | ", regla[2]," | ", regla[3]," | ", regla[4])

  etiqueta <- paste0("rL", 361 + i/21)

  cond_1_1 <- paste0("(B$v", 10063 + i, " == 1)")
  cond_2_1 <- paste0("(B$v", 10064 + i, " == 0 | is.na(B$v", 10064 + i, "))")
  cond_2_2 <- paste0("(B$v", 10065 + i, " == 0 | is.na(B$v", 10065 + i, "))")
  filtrado <- paste0(cond_1_1, " & ", cond_2_1, " & ", cond_2_2)
  AddRule(etiqueta, regla, filtrado)
}
rm(i,j,cond_1_1,cond_2_1,cond_2_2,regla,etiqueta,filtrado)
```

```{r}
# Cálculo del total de errores por empresa.
# G$DOCE <- NULL
vars_suma <- names(G)[!(names(G) %in% lista_vars_identif)]
Err_x_Col <- apply(G[, vars_suma], 2, sum, na.rm = T) # Errores por variable de validación.
G$SumERR <- apply(G[, vars_suma], 1, sum, na.rm = T) # Errores por empresa
```

```{r}
# Transformación a Excel del dataframe G que contiene las validaciones.
writexl::write_xlsx(G, "C:/Users/Documents/EMPRESAS 2020/VALIDAC_LOG_11ra_2020.xlsx")
```

--- FIN DE LA SINTAXIS DE VALIDACIÓN DEL MÓDULO AMBIENTAL DE LA ENESEM 2020. ---

ANEXO B. Código R Markdown para el control de integridad de la base de datos.

```
---
title: "Control de Integridad 2020"
author: "Ramiro Benavides"
date: "08/03/2022"
update: "08/03/2021"
output: html_document
---
```

1. Leer el libro de Excel de RESTRICCIONES_2020.

```
```{r}
RESTRICCIONES_2020 <- readxl::read_excel("~/EMPRESAS 2020/RESTRICCIONES_2020.xlsx")
```
```

2. Leer la BDD de ENESEM 2020 en formato SPSS.

```
```{r}
B <- haven::read_sav("~/EMPRESAS 2020/11er Corte - 07-03-2022/BDD/ENESEM 2020 - 07_03_2022.sav")
```
```

3. Carga de la base de datos en formato CSV. Se simula la variable "Zonal_DEAGA", en caso de ser necesario, hasta conocer la zonal a la cual pertenece el crítico.

```
```{r}
B_orig <- B

B <- B[which(B$estado == "C" & B$efectividad == 1), ]
# B <- EMP_2020[!is.na(EMP_2020$nombre_critico), ]

# Eliminar primero las empresas no efectivas (dejar únicamente a las efectivas).
B <- B[B$efectividad==1, ]

unique(B$nombre_critico)

# Eliminar a las empresas fusionadas y absorbidas.
B <- B[B$fusionada!="1.7" & B$absorbida!="1.8", ]

# Eliminar a la empresa TAME, que es efectiva y criticada, pero ya no existe.
B <- B[-which(B$inec_identificador_empresa == "41909487178"), ]
```
```

4. Creación de la variable 'Zonal_DEAGA' en base a información de críticos por zonal.

```
```{r}
# Asignar empresas a zonal, según el nombre del crítico.
B$Zonal_DEAGA <- NA_character_
B$Zonal_DEAGA[B$nombre_critico == "MUÑOZ MORA HARLETH ENYTH"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "CASTRO VILLALVA ROBERTO MAURICIO"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "SUMBA AYALA MARCIA IVON"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "AYALA SUMBA  MARCIA IVON"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "TERAN LOJA ADRIAN OSWALDO"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "VERGARA URGILES  REBECA"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "MALIZA CHASI LUZ ANGELICA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "YAUTIBUG BARRERA KATERIN PAOLA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "URGILÉS URGILÉS ENMA CUMANDÁ"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SANCHEZ RAMIREZ GERMAN ANDRES"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "MORETA SARAGURO EVELYN SOFIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "GONZALEZ PIONCE FABRIZIO ADRIAN"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "VALENCIA PEREZ FERNANDO DANIEL"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "BEDOYA PEÑAFIEL ANDRES PATRICIO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "BALSECA VILLALVA LESLIE PRISCILLA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "HOLGUIN QUIJIJE KENYS ELIZABETH"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "NIETO VERA MARYLIN KATHERINE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "VILLACIS VILLACIS VICTOR HUGO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "PEÑA ICAZA ARMANDO BERNARDINO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "BENITES CAMPOVERDE JESSICA DENISE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "RAMOS MUÑOZ JIREMY JACQUELINE"] <- "Litoral"
```
```

```
B$Zonal_DEAGA[B$nombre_critico == "ALVARADO CEDEÑO SELENE GARDENIA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "NACEVILLA CACHAGO MIREYA PATRICIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "GUEVARA ALVARADO ELICEO FRANCISCO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "GOMEZ MEJIA BYRON DAVID"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "VALDEZ SALAZAR JOSSELIN NOHELY"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SEVILLA VILLACIS CRISTINA SOLANGE"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "HERNANDEZ JATIVA MARIA PAULINA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "PAREDES ROMERO ALEXANDER JAVIER"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "CALIXTO FARIÑO JOHNNY ALEXIS"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "SEMANATE CAJIAO ANTONIO JAVIER"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SEMANATE ALVAREZ LEONELA DE LOS ANGELES"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "TOMALA MIRANDA STEFANIE YESENIA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "RODRIGUEZ BENAVIDES BRAYAN VLADIMIR"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "RUANO VACA CRISTINA ELIZABETH"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "CALLE PALACIOS MAGALY CARLOTA"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "GUARDERAS NÚÑEZ TATIANA VALERIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "PICO SANCHEZ ROSA PATRICIA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "MAIGUA VELA MAGDALENA DEL CONSUELO"] <- "DICA"
```

5. Crear el dataframe con las variables de identificación, el cual irá agregando después las variables de validación. Este frame se enviará a las zonales para su validación.

```{r}
lista_vars_identif <- c("inec_zonal", "Zonal_DEAGA", "inec_tamanio", "inec_ciiu4", "inec_identificador_empresa", "novedad",
                "efectividad", "ruc_muestra", "ruc", "nombre_comercial", "razon_social", "desc_actividad_principal",
                "ciiu4_actividad_principal", "desc_actividad_secundaria", "ciiu4_actividad_secundaria", "nombre_critico",
"fecha_critica")
G <- subset(B, select = lista_vars_identif)
G_orig <- G
```

6. Funciones utilitarias.

```{r}
AddRule <- function(label, regla, filtro="") {
  comando1 <- paste0("K <- dim(B)[1]")
  comando2 <- paste0("K <- length(which(", filtro, "))")
  comando <- ifelse(filtro == "", comando1, comando2)
  eval(parse(text = comando), envir = .GlobalEnv)
  M <- get0("K")
  r <- get0("G")
  comando1 <- paste0("r$", label, " <- ifelse(!(", regla, "), 1, NA)")
  comando2 <- paste0("r$", label, " <- ifelse(!(", regla, ") & (", filtro, "), 1, NA)")
  comando <- ifelse(filtro == "", comando1, comando2)
  eval(parse(text = comando))
  if (!(all(is.na(r[dim(r)[2]])))) {
     assign("G", r, envir = .GlobalEnv)
     N <- sum(r[dim(r)[2]], na.rm = T)
     print(paste0("SÍ se agregó la regla ", label, " al frame de validación. ",
                  "N/M=", N, "/", M))
  } else {
    print(paste0("NO se agregó la regla ", label, " al frame de validación"))
  }
  comando <- paste0("rm(K)")
  eval(parse(text = comando), envir = .GlobalEnv)
}
```

7. Creación de variables numéricas auxiliares en B.

```{r}
B$UNO <- 1
B$CIEN <- 100
```

8. Verificación de integridad para todas las variables numéricas.

```{r}
for (i in 1:dim(RESTRICCIONES_2020)[1]) {
  vlabel <- paste0('c', substr(RESTRICCIONES_2020$V_2019[i], 2, nchar(RESTRICCIONES_2020$V_2019[i])))
```

```
  AddRule(vlabel, RESTRICCIONES_2020$CONDICION[i], RESTRICCIONES_2020$RESTRICCION[i])
}
```

9. Determinación de total de errores por empresa y por variable de control.

```{r}
# Cálculo del total de errores por empresa.
# G$DOCE <- NULL
vars_suma <- names(G)[!(names(G) %in% lista_vars_identif)]
Err_x_Col <- apply(G[, vars_suma], 2, sum, na.rm = T) # Errores por variable de validación.
G$SumERR <- apply(G[, vars_suma], 1, sum, na.rm = T) # Errores por empresa
```

10. Reporte del frame de validación.

```{r}
writexl::write_xlsx(G, "~/EMPRESAS 2020/Control de integridad 2020.xlsx")
```

11. Corrección de los valores erróneos en todas las variables numéricas del frame B.

```{r}
# Creación de un nuevo frame "C" con los valores corregidos a partir de B.
C <- B

# Corrección de los errores en orden alfanumérico de aparición.
# Corrección del error c711.
table(B$v711)
C$v711[is.na(B$v711)] <- 2
table(C$v711)

# Corrección del error c713.
table(B$v713)
C$v713[is.na(B$v713)] <- 2
table(C$v713)

# Corrección del error c714.
table(B$v714)
C$v714[is.na(B$v714)] <- 2
table(C$v714)

# Corrección del error c715.
table(B$v715)
C$v715[is.na(B$v715)] <- 2
table(C$v715)

# Corrección del error c9ii1.
table(B$v9ii1)
C$v9ii1[is.na(B$v9ii1)] <- 2
table(C$v9ii1)

# Corrección del error c10002.
# No se corrige esta variable, porque es la observación de agua de red pública,
# y al final se eliminarán de la BDD de publicación TODAS las observaciones.

# Corrección del error c10006.
table(B$v10006)
C$v10006[which(G$c10006 == 1)] <- 2
C$v10007[which(G$c10006 == 1)] <- NA_integer_
C$v10008[which(G$c10006 == 1)] <- NA_integer_
C$v10012[which(G$c10006 == 1)] <- NA_integer_
table(C$v10006)

# Corrección del error c10006.
table(B$v10006)
C$v10006[which(G$c10006 == 1)] <- 2
C$v10007[which(G$c10006 == 1)] <- NA_integer_
C$v10008[which(G$c10006 == 1)] <- NA_integer_
C$v10012[which(G$c10006 == 1)] <- NA_integer_
```

```
table(C$v10006)

# Corrección del error c10022.
table(B$v10022)
C$v10022[which(G$c10022 == 1)] <- 2
C$v10023[which(G$c10022 == 1)] <- NA_integer_
C$v10024[which(G$c10022 == 1)] <- NA_integer_
C$v10028[which(G$c10022 == 1)] <- NA_integer_

C$v10030[which(B$v10030 == 0)] <- NA_real_
C$v10031[which(B$v10031 == 0)] <- NA_real_

table(C$v10022)

# Corrección del error c10ii6.
table(B$v10ii6)
C$v10ii6[which(G$c10ii6 == 1)] <- 100
C$v10036[which(G$c10ii6 == 1)] <- C$v10049[which(G$c10ii6 == 1)]
C$v10037[which(G$c10ii6 == 1)] <- C$v10050[which(G$c10ii6 == 1)]

C$v10038[which(G$c10ii6 == 1)] <- C$v10051[which(G$c10ii6 == 1)]
C$v10039[which(G$c10ii6 == 1)] <- C$v10052[which(G$c10ii6 == 1)]

C$v10040[which(G$c10ii6 == 1)] <- C$v10053[which(G$c10ii6 == 1)]
C$v10041[which(G$c10ii6 == 1)] <- C$v10054[which(G$c10ii6 == 1)]

C$v10042[which(G$c10ii6 == 1)] <- C$v10055[which(G$c10ii6 == 1)]
C$v10043[which(G$c10ii6 == 1)] <- C$v10056[which(G$c10ii6 == 1)]

C$v10044[which(G$c10ii6 == 1)] <- C$v10057[which(G$c10ii6 == 1)]
C$v10045[which(G$c10ii6 == 1)] <- C$v10058[which(G$c10ii6 == 1)]

C$v10046[which(G$c10ii6 == 1)] <- C$v10059[which(G$c10ii6 == 1)]
C$v10047[which(G$c10ii6 == 1)] <- C$v10060[which(G$c10ii6 == 1)]

C$v10048[which(G$c10ii6 == 1)] <- C$v10061[which(G$c10ii6 == 1)]

C$v10049[which(G$c10ii6 == 1)] <- NA_integer_
C$v10050[which(G$c10ii6 == 1)] <- NA_integer_
C$v10051[which(G$c10ii6 == 1)] <- NA_integer_
C$v10052[which(G$c10ii6 == 1)] <- NA_integer_
C$v10053[which(G$c10ii6 == 1)] <- NA_integer_
C$v10054[which(G$c10ii6 == 1)] <- NA_integer_
C$v10055[which(G$c10ii6 == 1)] <- NA_integer_
C$v10056[which(G$c10ii6 == 1)] <- NA_integer_
C$v10057[which(G$c10ii6 == 1)] <- NA_integer_
C$v10058[which(G$c10ii6 == 1)] <- NA_integer_
C$v10059[which(G$c10ii6 == 1)] <- NA_integer_
C$v10060[which(G$c10ii6 == 1)] <- NA_integer_
C$v10061[which(G$c10ii6 == 1)] <- NA_integer_
table(C$v10ii6)

# Corrección del error c10036.
table(B$v10036)
C$v10036[which(G$c10036 == 1)] <- 2
table(C$v10036)

# Corrección del error c10038.
table(B$v10038)
C$v10038[which(G$c10038 == 1)] <- 2
table(C$v10038)

# Corrección del error c10040.
table(B$v10040)
C$v10040[which(G$c10040 == 1)] <- 2
table(C$v10040)
# Corrección del error c10042.
```

```
table(B$v10042)
C$v10042[which(G$c10042 == 1)] <- 2
table(C$v10042)

# Corrección del error c10044.
table(B$v10044)
C$v10044[which(G$c10044 == 1)] <- 2
table(C$v10044)

# Corrección del error c10046.
table(B$v10046)
C$v10046[which(G$c10046 == 1)] <- 2
table(C$v10046)

# Corrección del error c10049.
table(B$v10049)
C$v10049[which(G$c10049 == 1)] <- 2
table(C$v10049)

# Corrección del error c10051.
table(B$v10051)
C$v10051[which(G$c10051 == 1)] <- 2
table(C$v10051)

# Corrección del error c10053.
table(B$v10053)
C$v10053[which(G$c10053 == 1)] <- 2
table(C$v10053)

# Corrección del error c10055.
table(B$v10055)
C$v10055[which(G$c10055 == 1)] <- 2
table(C$v10055)

# Corrección del error c10057.
table(B$v10057)
C$v10057[which(G$c10057 == 1)] <- 2
table(C$v10057)

# Corrección del error c10059.
table(B$v10059)
C$v10059[which(G$c10059 == 1)] <- 2
table(C$v10059)

# Corrección del error c10063.
table(B$v10063)
C$v10063[which(G$c10063 == 1)] <- 2
table(C$v10063)

# Corrección de los acarreos de la v10062 por corrección de la c10063.
table(B$v10062)
C$v10062[which(C$v10063 > 0)] <- 1
table(C$v10062)

# Corrección de un caso específico de combinación de variables:
# v10062, v10063, v10064, v10065, v10080.
C$v10063[which(B$v10062 == 1 & B$v10064 == 1 & B$v10065 == 845 & B$v10080 == 100)] <- 1

# # Corrección integral de la Línea 361 (Chatarra liviana), cabecera = v10062.
# cond_0 <- with(B, v10062 == 1 & v10063 == 1 & v10064 %in% 1:2 & v10065 == 0)
# cond_1 <- with(B, v10062 == 1 & v10063 == 1 & is.na(v10064))
# cond_2 <- with(B, v10062 == 1 & v10063 == 1 & is.na(v10064) & is.na(v10075) & is.na(v10076))
# cond_3 <- with(B, v10062 == 1 & v10063 == 1 & v10064 == 1)
# cond_4 <- with(B, v10062 == 1 & v10063 == 2)
# cond_5 <- with(B, v10062 == 1 & v10063 == 2 & v10076 == 630)
# cond_6 <- with(B, v10062 == 1 & v10063 == 1 & is.na(v10064) & v10065 == 2)
# cond_7 <- with(B, v10062 == 1 & v10063 == 2 & v10074 == 0)
```

```
# cond_8 <- with(B, v10062 == 1 & v10063 == 1)
# cond_9 <- with(B, v10062 == 2)
#
# C$v10064[cond_0] <- NA_integer_
# C$v10065[cond_0] <- NA_real_
#
# C$v10065[cond_1] <- NA_real_
# C$v10067[cond_1] <- NA_real_
# C$v10069[cond_1] <- NA_real_
# C$v10071[cond_1] <- NA_real_
# C$v10073[cond_1] <- NA_real_
# C$v10074[cond_1] <- NA_real_
#
# C$v10065[cond_2] <- NA_real_
# C$v10067[cond_2] <- NA_real_
# C$v10069[cond_2] <- NA_real_
# C$v10071[cond_2] <- NA_real_
# C$v10073[cond_2] <- NA_real_
# C$v10074[cond_2] <- NA_real_
# C$v10075[cond_2] <- 2
# C$v10077[cond_2] <- 2
#
# C$v10079[is.na(C$v10079[cond_3])] <- 0
# C$v10080[is.na(C$v10080[cond_3])] <- 0
# C$v10081[is.na(C$v10081[cond_3])] <- 0
#
# C$v10070[cond_4] <- 2
# C$v10072[cond_4] <- 1
# C$v10073[cond_4] <- NA_real_
# C$v10074[cond_4] <- NA_real_
# C$v10075[cond_4] <- 2
# C$v10077[cond_4] <- 2
# vars_suma <- c("v10079", "v10080", "v10081")
# SumGestion <- apply(B[cond_4, vars_suma], 1, sum, na.rm = T) # Porcentaje acumulado Gestión Residuos.
# C$v10079[which(SumGestion != 100)] <- 100
#
# C$v10075[cond_5] <- 1
# C$v10081[cond_5] <- 100
#
# C$v10065[cond_6] <- NA_real_
#
# C$v10070[cond_7] <- 2
# C$v10072[cond_7] <- 1
# C$v10074[cond_7] <- NA_real_
# C$v10075[cond_7] <- 2
# C$v10077[cond_7] <- 2
# C$v10079[cond_7] <- 100
#
# C$v10079[is.na(C$v10079[cond_8])] <- 0
# C$v10080[is.na(C$v10080[cond_8])] <- 0
# C$v10081[is.na(C$v10081[cond_8])] <- 0
#
# C[cond_9, 2766:2784] <- NA
```

12. Reporte del frame íntegro.

```{r}
haven::write_sav(C, "~/EMPRESAS 2020/BDD Integra.sav")
```

ANEXO C. Código R Markdown para las validaciones especiales.

```
---
title: "Validaciones especiales 2020"
author: "Ramiro Benavides"
date: "07/10/2021"
output: html_document
---
```

1. Carga de la base de datos ENESEM 2020 en versión Excel.

```
```{r}
# BDD_8vo <- readxl::read_excel("~/EMPRESAS 2020/8vo Corte - 05-11-2021/BDD/BDD_8vo_05_11_2021.xlsx")
# NOTA: Se recomienda trabajar cargando la versión SPSS del archivo anterior, para evitar caracteres extraños.
# Intersección de identificadores de la BDD global con la base efectiva y con críticos.
id <- intersect(B_2020$inec_identificador_empresa, BDD_9no$inec_identificador_empresa)
ind <- which(BDD_9no$inec_identificador_empresa %in% id)
```
```

2. Generación de la BDD de trabajo.

```
```{r}
observacion <- NA_character_
BDD <- cbind(B_2020_orig, observacion)
BDD <- BDD[which(BDD$nombre_critico != "" & BDD$estado == "C"), ]
BDD <- BDD[BDD$fusionada!="1.7" & BDD$absorbida!="1.8", ]
BDD <- BDD[BDD$efectividad == 1, ]
BDD <- BDD[which(BDD$inec_identificador_empresa %in% id), ]
```
```

3. Generación de las diferentes hojas de Energía del libro "Reporte Otros Usos Energía y Combustibles_BDD_30_09_2021.xlsx".

```
```{r}
# Usar la lista de variables "lista_vars_identif" de la corrida de comparaciones.
lista_vars_identif <- c("inec_zonal", "Zonal_DEAGA", "inec_tamanio", "inec_ciiu4", "inec_identificador_empresa", "novedad",
                        "efectividad", "ruc_muestra", "ruc", "nombre_comercial", "razon_social", "desc_actividad_principal",
                        "ciiu4_actividad_principal", "desc_actividad_secundaria", "ciiu4_actividad_secundaria", "nombre_critico",
"fecha_critica")

E_Solar <- BDD[ind, c(lista_vars_identif, "v9004", "v9008", "v9011", "observacion")]
E_Solar <- E_Solar[which(E_Solar$v9004 == 1 & E_Solar$v9008 == 3), ]

E_Eolica <- BDD[ind, c(lista_vars_identif, "v9012", "v9016", "v9019", "observacion")]
E_Eolica <- E_Eolica[which(E_Eolica$v9012 == 1 & E_Eolica$v9016 == 3), ]

E_Biomasa <- BDD[ind, c(lista_vars_identif, "v9020", "v9024", "v9027", "observacion")]
E_Biomasa <- E_Biomasa[which(E_Biomasa$v9020 == 1 & E_Biomasa$v9024 == 3), ]

E_Hidraulica <- BDD[ind, c(lista_vars_identif, "v9028", "v9032", "v9035", "observacion")]
E_Hidraulica <- E_Hidraulica[which(E_Hidraulica$v9028 == 1 & E_Hidraulica$v9032 == 3), ]

E_Generador <- BDD[ind, c(lista_vars_identif, "v9036", "v9040", "v9043", "observacion")]
E_Generador <- E_Generador[which(E_Generador$v9036 == 1 & E_Generador$v9040 == 3), ]

E_Otro <- BDD[ind, c(lista_vars_identif, "v9057", "v9044", "v9048", "v9051", "observacion")]
E_Otro <- E_Otro[which(E_Otro$v9044 == 1 & E_Otro$v9048 == 3), ]
```
```

4. Generación de las diferentes hojas de Combustibles del libro "Reporte Otros Usos Energía y Combustibles_BDD_30_09_2021.xlsx".

```
```{r}
# Usar la lista de variables "lista_vars_identif" de la corrida de comparaciones.
G_Super <- BDD[ind, c(lista_vars_identif, "v9060", "v9061", "observacion")]
G_Super <- G_Super[which(G_Super$v9060 == 6), ]

G_Extra <- BDD[ind, c(lista_vars_identif, "v9064", "v9065", "observacion")]
G_Extra <- G_Extra[which(G_Extra$v9064 == 6), ]

G_Jet <- BDD[ind, c(lista_vars_identif, "v9068", "v9069", "observacion")]
```
```

```
G_Jet <- G_Jet[which(G_Jet$v9068 == 6), ]

G_Diesel <- BDD[ind, c(lista_vars_identif, "v9072", "v9073", "observacion")]
G_Diesel <- G_Diesel[which(G_Diesel$v9072 == 6), ]

G_GLP <- BDD[ind, c(lista_vars_identif, "v9076", "v9077", "observacion")]
G_GLP <- G_GLP[which(G_GLP$v9076 == 6), ]

G_Gas <- BDD[ind, c(lista_vars_identif, "v9080", "v9081", "observacion")]
G_Gas <- G_Gas[which(G_Gas$v9080 == 6), ]

G_Fuel <- BDD[ind, c(lista_vars_identif, "v9084", "v9085", "observacion")]
G_Fuel <- G_Fuel[which(G_Fuel$v9084 == 6), ]

G_Crudo <- BDD[ind, c(lista_vars_identif, "v9088", "v9089", "observacion")]
G_Crudo <- G_Crudo[which(G_Crudo$v9088 == 6), ]

G_Carbon <- BDD[ind, c(lista_vars_identif, "v9092", "v9093", "observacion")]
G_Carbon <- G_Carbon[which(G_Carbon$v9092 == 6), ]

G_Ecopais <- BDD[ind, c(lista_vars_identif, "v9096", "v9097", "observacion")]
G_Ecopais <- G_Ecopais[which(G_Ecopais$v9096 == 6), ]

G_Aceites <- BDD[ind, c(lista_vars_identif, "v9100", "v9101", "observacion")]
G_Aceites <- G_Aceites[which(G_Aceites$v9100 == 6), ]

G_Otros <- BDD[ind, c(lista_vars_identif, "otro_combustible", "v9103", "v9104", "observacion")]
G_Otros <- G_Otros[which(G_Otros$v9103 == 6), ]
```

5. Reporte del libro "Reporte Otros Usos Energía y Combustibles_BDD_30_09_2021.xlsx".

```{r}
writexl::write_xlsx(list("Otros usos - E. Solar" = E_Solar,
                         "Otros usos - E. Eólica" = E_Eolica,
                         "Otros usos - E. Biomasa" = E_Biomasa,
                         "Otros usos - E. Hidráulica" = E_Hidraulica,
                         "Otros usos - G. Termoeléctrico" = E_Generador,
                         "Otros usos - Otro tipo de energía" = E_Otro,
                         "Otros usos - Gasolina Súper" = G_Super,
                         "Otros usos - Gasolina Extra" = G_Extra,
                         "Otros usos - Jet Fuel" = G_Jet,
                         "Otros usos - Diesel" = G_Diesel,
                         "Otros usos - GLP" = G_GLP,
                         "Otros usos - Gas natural" = G_Gas,
                         "Otros usos - Residuo Fuel Oil" = G_Fuel,
                         "Otros usos - Crudo residual" = G_Crudo,
                         "Otros usos - Carbón" = G_Carbon,
                         "Otros usos - Gasolina Ecopaís" = G_Ecopais,
                         "Otros usos - Aceites" = G_Aceites,
                         "Otros usos - Otros combustibles" = G_Otros),
                    "~/EMPRESAS 2020/9no Corte - 25-11-2021/Validaciones especiales/Reporte Otros Usos Energía y Combustibles_BDD_25_11_2021.xlsx")
```

6. Generación de las diferentes hojas de reporte del libro "Reporte Otros Capítulos 7 y 9_BDD_30_09_2021.xlsx".

```{r}
# Usar la lista de variables "lista_vars_identif" de la corrida de comparaciones.
Reporte_7A <- BDD[ind, c(lista_vars_identif, "v7024", "v7025", "observacion")]
Reporte_7A <- Reporte_7A[which(Reporte_7A$v7024 > 0), ]

Reporte_7B <- BDD[ind, c(lista_vars_identif, "v75", "v751", "v7511", "observacion")]
Reporte_7B <- Reporte_7B[which(Reporte_7B$v75 == 1 & Reporte_7B$v751 == 4), ]

Reporte_9A <- BDD[ind, c(lista_vars_identif, "v9044", "v9057", "v9051", "observacion")]
Reporte_9A <- Reporte_9A[which(Reporte_9A$v9044 == 1), ]
```

7. Reporte del libro "Reporte Otros Capítulos 7 y 9_BDD_30_09_2021.xlsx".

```{r}
writexl::write_xlsx(list("Reporte 7A" = Reporte_7A,
                         "Reporte 7B" = Reporte_7B,
                         "Reporte 9A" = Reporte_9A),
                    "~/EMPRESAS 2020/9no Corte - 25-11-2021/Validaciones especiales/Reporte Otros Capítulos 7 y 9_BDD_25_11_2021.xlsx")
```

ANEXO D. Código R Markdown para las comparaciones de valores 2020-2019.

```
---
title: "SX Comparación BDD 2020-2019"
author: "Ramiro Benavides"
date: "03/08/2021"
output: html_document
---
```

A. Cargar las BDD 2020 y 2019 del Módulo Ambiental ENESEM.

```
```{r}
B_2020 <- haven::read_sav("~/EMPRESAS 2020/9no Corte - 25-11-2021/BDD/Empresasfusionado.sav")
B_2019 <- haven::read_sav("~/EMPRESAS 2019/Publicación/Documento Técnico/BDD_AMB_a_PUBLICAR_2019_W.sav")

# B_2020 <- B_2020[which(!is.na(B_2020$secuencial_sistema)), ]
```
```

B. Agregado de la variable "Zonal_DEAGA" a la BDD 2020.

```
```{r}
B_2020$Zonal_DEAGA <- NA_character_
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "MUÑOZ MORA HARLETH ENYTH"] <- "Centro"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "CASTRO VILLALVA ROBERTO MAURICIO"] <- "Centro"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "SUMBA AYALA MARCIA IVON"] <- "Sur"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "AYALA SUMBA  MARCIA IVON"] <- "Sur"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "TERAN LOJA ADRIAN OSWALDO"] <- "Sur"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "VERGARA URGILES  REBECA"] <- "Sur"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "MALIZA CHASI LUZ ANGELICA"] <- "Centro"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "YAUTIBUG BARRERA KATERIN PAOLA"] <- "Centro"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "URGILÉS URGILÉS ENMA CUMANDÁ"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "SANCHEZ RAMIREZ GERMAN ANDRES"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "MORETA SARAGURO EVELYN SOFIA"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "GONZALEZ PIONCE FABRIZIO ADRIAN"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "VALENCIA PEREZ FERNANDO DANIEL"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "BEDOYA PEÑAFIEL ANDRES PATRICIO"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "BALSECA VILLALVA LESLIE PRISCILLA"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "HOLGUIN QUIJIJE KENYS ELIZABETH"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "NIETO VERA MARYLIN KATHERINE"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "VILLACIS VILLACIS VICTOR HUGO"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "PEÑA ICAZA ARMANDO BERNARDINO"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "BENITES CAMPOVERDE JESSICA DENISE"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "RAMOS MUÑOZ JIREMY JACQUELINE"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "ALVARADO CEDEÑO SELENE GARDENIA"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "NACEVILLA CACHAGO MIREYA PATRICIA"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "GUEVARA ALVARADO ELICEO FRANCISCO"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "GOMEZ MEJIA BYRON DAVID"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "VALDEZ SALAZAR JOSSELIN NOHELY"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "SEVILLA VILLACIS CRISTINA SOLANGE"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "HERNANDEZ JATIVA MARIA PAULINA"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "PAREDES ROMERO ALEXANDER JAVIER"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "CALIXTO FARIÑO JOHNNY ALEXIS"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "SEMANATE CAJIAO ANTONIO JAVIER"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "SEMANATE ALVAREZ LEONELA DE LOS ANGELES"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "TOMALA MIRANDA STEFANIE YESENIA"] <- "Litoral"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "RODRIGUEZ BENAVIDES BRAYAN VLADIMIR"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "RUANO VACA CRISTINA ELIZABETH"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "CALLE PALACIOS MAGALY CARLOTA"] <- "Sur"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "GUARDERAS NÚÑEZ TATIANA VALERIA"] <- "DICA"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "PICO SANCHEZ ROSA PATRICIA"] <- "Centro"
B_2020$Zonal_DEAGA[B_2020$nombre_critico == "MAIGUA VELA MAGDALENA DEL CONSUELO"] <- "DICA"

# B_2020$Zonal_DEAGA[which(is.na(B_2020$Zonal_DEAGA) & B_2020$efectividad == 1 & B_2020$inec_zonal == 2)] <- "Centro"
# B_2020$Zonal_DEAGA[which(is.na(B_2020$Zonal_DEAGA) & B_2020$efectividad == 1 & B_2020$inec_zonal == 3)] <- "Litoral"
# B_2020$Zonal_DEAGA[which(is.na(B_2020$Zonal_DEAGA) & B_2020$efectividad == 1 & B_2020$inec_zonal == 4)] <- "DICA"
# B_2020$Zonal_DEAGA[which(is.na(B_2020$Zonal_DEAGA) & B_2020$efectividad == 1 & B_2020$inec_zonal == 5)] <- "Sur"
```
```

C. Depurar la BDD 2020 (quitando variables irrelevantes).

```{r}
B_2020_orig <- B_2020
ind <- which(B_2020$fusionada == "1.7" | B_2020$absorbida == "1.8")
B_2020 <- B_2020[-ind, ]
B_2020 <- B_2020[which(B_2020$efectividad == 1), ]
B_2020 <- B_2020[-which(B_2020$inec_identificador_empresa == "41909487178"), ] # id_empresa de TAME.

lista_vars_identif <- c("inec_zonal", "Zonal_DEAGA", "inec_tamanio", "inec_ciiu4", "inec_identificador_empresa", "novedad",
            "efectividad", "ruc_muestra", "ruc", "nombre_comercial", "razon_social", "desc_actividad_principal",
            "ciiu4_actividad_principal", "desc_actividad_secundaria", "ciiu4_actividad_secundaria", "nombre_critico",
"fecha_critica")

lista_vars_DEAGA <- names(B_2020)[2444:3736]
lista_vars <- c(lista_vars_identif, lista_vars_DEAGA)
B_2020 <- B_2020[, lista_vars]

rango_vars_control <- c(19:21, 25:26, 30, 32, 55:56, 65, seq(67, 71, 2), seq(73, 77, 2),
                        111, 113, seq(115, 119, 2), 156:158, 159:160, 211:215, 218:219,
                        222:223, 230:231, 266, 267:268, 273:274, 301, 303, 308, 309:313,
                        344, 349, 407, 412, 428, 433, 533, 538, 656, 660, 665, 681, 686,
                        702, 707, 807, 812)

rango_vars_control <- rango_vars_control - 1
lista_vars_control <- names(B_2020)[rango_vars_control]
lista_vars <- c(lista_vars_identif, lista_vars_control)


B_2020 <- subset(B_2020, select = lista_vars)
# ind <- which(B_2020$inec_identificador_empresa == "14802259175") # Esta empresa ha sido absorbida,
                                                                   # y no tiene información ambiental.
rm(ind)
```

D. Seleccionar las variables que serán comparadas entre las BDD 2020 y 2019.

```{r}
# rango_vars_encuesta <- c(19:21, 25:26, 30, 32, 55:56, 65, seq(67, 71, 2), seq(73, 77, 2),
#                          111, 113, seq(115, 119, 2), 156:158, 159:160, 211:215, 218:219,
#                          222:223, 230:231, 266, 267:268, 273:274, 301, 303, 308, 309:313,
#                          344, 349, 407, 412, 428, 433, 533, 538, 656, 660, 665, 681, 686,
#                          702, 707, 807, 812)
# lista_vars_encuesta <- names(B_2020)[rango_vars_encuesta]
# lista_vars <- c(lista_vars_identif, lista_vars_encuesta)
```

E. Creación de BDD 2019 y 2020 listas para someterse a comparación caso a caso. Años 2019 y 2020.

```{r}
C <- intersect(B_2019$inec_identificador_empresa, B_2020$inec_identificador_empresa)
empresas_comunes <- B_2020$inec_identificador_empresa[which(B_2020$inec_identificador_empresa %in% C & B_2020$efectividad == 1)]
# A es la base año 2019 que contiene a las empresas efectivas comunes en ambos años 2019 y 2020.
A <- B_2019[B_2019$inec_identificador_empresa %in% empresas_comunes, ]
# B es la base año 2020 que contiene a las empresas efectivas comunes en ambos años 2019 y 2020.
B <- B_2020[B_2020$inec_identificador_empresa %in% empresas_comunes, ]
# B <- B[-c(220,286), ] # Se eliminan las filas 220 y 286 por duplicidad de identificador de empresa.
u <- empresas_comunes
rm(empresas_comunes, C)
```

F. Ajuste del tipo de datos para la BDD 2020, según los tipos de datos de las variables correspondientes de la BDD 2019.

```{r}
A <- data.frame(A)
```

G. Creación de los frames de cabecera (variables de identificación) y reporte.

```{r}
cabecera <- B[, 1:17]
```

```
datos <- data.frame(matrix(NA, nrow=length(u), ncol=6*length(lista_vars_control)))
names(datos)[2 + seq(1, 6*length(lista_vars_control) - 3, 6)] <- lista_vars_control

# reporte es el frame que junta cabecera+datos 2019 y 2020 de las variables a contrastar.
reporte <- data.frame(matrix(NA, nrow=length(u), ncol=6*length(lista_vars_control) + 17))
names(reporte) <- c(names(cabecera), names(datos))
```

H. Comparación caso a caso (según id_empresa) de las variables 2019 y 2020.

```{r}
Nivel_OK <- rep(NA_real_, dim(B)[1])  # Vector de grados de correción de empresas.
rango_var_almacen <- c(10070, 10133, 10154, 10259, 10385, 10406, 10427, 10532) # Variables de
# almacenamiento de residuos no peligrosos y especiales (las de mayor cantidad).
lista_var_almacen <- paste0("v", rango_var_almacen)
ind <- which(lista_vars_control %in% lista_var_almacen) # Vector de índices de las variables de
# almacenamiento de residuos no peligrosos y especiales, según la lista de variables de control.

for (q in 1:dim(B)[1]) {
  filaA <- which(A$inec_identificador_empresa == u[q])
  filaB <- q

  # Llenado de variables de identificación del dataframe "reporte".
  reporte[q, 1:17] <- cabecera[q, 1:17]

  E <- NA_real_ # Inicialización del vector útil para cálculo de distribución empírica de variaciones interanuales.

  # LLenado de datos años 2019 y 2020 en el dataframe "reporte"
  for (k in 1:length(lista_vars_control)) {
    reporte[q, 12 + 6*k] <- as.numeric(A[filaA, lista_vars_control[k]]) # Grabar dato año 2019.
    reporte[q, 13 + 6*k] <- as.numeric(B[q, lista_vars_control[k]]) # Grabar dato año 2020.

    if (isFALSE(k %in% ind)) { # En caso de que el índice de columna no esté en "ind".
      tempo <- ifelse(is.na(reporte[q, 13 + 6*k]) & (reporte[q, 12 + 6*k] > 0), 1,
                      abs(reporte[q, 13 + 6*k] - reporte[q, 12 + 6*k]) / reporte[q, 12 + 6*k])
    }
    if (isTRUE(k %in% ind)) { # En caso de que el índice de columna sí esté en "ind".
      tempo <- ifelse((reporte[q, 12 + 6*k]==1 & reporte[q, 13 + 6*k]==1), 1, 0)
    }

    tempo <- ifelse(is.nan(tempo), NA_real_, tempo)
    reporte[q, 14 + 6*k] <- tempo   # Grabar variación interanual.
    E <- c(E, reporte[q, 14 + 6*k])   # E es el vector de diferencia relativa
                                      # interanual entre los datos del caso "q".
  }

  E <- E[-1] # Eliminación del NA inicial.
  names(E) <- lista_vars_control

  # E1 es el frame que sirve para realizar los cálculos de estadígrafos de la
  # distribución de porcentajes (proporciones) de cambios interanuales por empresa.
  E1 <- data.frame(E[which(E >= 0)])

  # F1 es la conversión de E1 desde dataframe a matriz.
  F1 <- as.matrix(E1)

  # G1 es la conversión de F1 de matriz a vector numérico.
  G1 <- F1[, 1]

  # Q es la distribución de centiles de los porcentajes de cambios interanuales por empresa.
  Q <- quantile(E1, seq(0.01, 1, 0.01), na.rm=T)

  # F es la función de distribución de probabilidad acumulada (empírica) de los
  # porcentajes de cambio interanuales por empresa.
  F <- ecdf(as.numeric(G1))

  # LOS VALORES IMPORTANTES A OBSERVAR EN LA FUNCIÓN F SON:
  # (1) El valor F(0), que devuelve la máxima proporción de variables numéricas con 0% de cambios interanuales.
```

```
  # (2) El valor F(0.3), que devuelve la máxima proporción de variables numéricas con 30% o menos de cambios interanuales.
  # (3) El valor F(1), que devuelve la máxima proporción de variables numéricas con 100% o menos de cambios interanuales.

  # De todos estos valores, el más importante es F(0.3), puesto que el umbral del 30% ha sido fijado en otros años como el valor máximo de cambio interanual que pueden sufrir las variables numéricas del Módulo Ambiental de la ENESEM. En este año 2020, se corrobora que el umbral del 30% tiene asidero en el tercer cuartil de la distribución empírica de la mayoría de variables del módulo.

  # Nivel_OK es el porcentaje de variables con cambio interanual menor o igual a 30%.
  Nivel_OK[q] <- F(0)*100

  cat("Caso", as.character(q), "subido con éxito\n")
}
rm(q,k,filaA,filaB,E,E1,F1,G1,F,Q,tempo,ind)
```

I. Inclusión al final del dataframe "reporte" de la columna "Nivel_OK".

```{r}
reporte <- cbind(reporte, Nivel_OK)
```

J. Exportación a Excel del dataframe de reporte.

```{r}
writexl::write_xlsx(reporte, "~/Documents/EMPRESAS 2020/2do Corte - 16-08-2021/Justificaciones zonales/Reporte.xlsx")
```

K. Generación de vector de recuento de variables sobre el umbral, por empresa.

```{r}
# recuento_err <- rep(NA_real_, dim(B)[1])
# for (i in 1:dim(B)[1]) {
#   d <- as.numeric(reporte[i, lista_vars_control])
#   recuento_err[i] <- length(which(d > 0.3 & !is.infinite(d)))
# }
# rm(i,d)
#   writexl::write_xlsx(data.frame(recuento_err), "~/Documents/EMPRESAS 2020/2do Corte - 16-08-2021/Justificaciones zonales/Vector_Recuento_Err.xlsx")
```

L. OPCIONAL: Generación de un vector de fechas, en el caso en que no se copien bien a los frames "reporte" y "datos" que se exportaron a Excel.

```{r}
fechas <- B$fecha_critica
writexl::write_xlsx(data.frame(fechas), "~/Documents/EMPRESAS 2020/9no Corte - 25-11-2021/vector_fechas_9na.xlsx")
```

M. Función de borrado de celdas adyacentes a una celda de coordenadas dadas (id_emp, variable).

```{r}
BorrarCeldas <- function(BDD, id_emp, variable) {
  comando1 <- paste0("ind_col <- which(names(", BDD, ") %in% '", variable, "')")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0("ind_fil <- which(", BDD, "$inec_identificador_empresa %in% '", id_emp, "')")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0(BDD, "[", ind_fil, ", ", ind_col - 2, "] <- NA")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0(BDD, "[", ind_fil, ", ", ind_col - 1, "] <- NA")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0(BDD, "[", ind_fil, ", ", ind_col, "] <- NA")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0(BDD, "[", ind_fil, ", ", ind_col + 1, "] <- NA")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0(BDD, "[", ind_fil, ", ", ind_col + 2, "] <- NA")
  eval(parse(text = comando1), envir = .GlobalEnv)
```

```
  comando1 <- paste0(BDD, "[", ind_fil, ", ", ind_col + 3, "] <- NA")
  eval(parse(text = comando1), envir = .GlobalEnv)

  comando1 <- paste0("rm(ind_fil, ind_col)")
  eval(parse(text = comando1), envir = .GlobalEnv)
}
```

N. Rutina para cargar un Excel con justificaciones de zonales y dejar únicamente los códigos 1 y 2.

```{r}
Z <- Sur_8 # En el frame "Z" se carga el Excel de la respuesta de la zonal que se quiera cargar.
id <- Z$inec_identificador_empresa
nom_vars <- names(Z)
ind_just <- grep("^Just", names(Z))
Z <- Z[, ind_just]
names(Z) <- nom_vars[ind_just - 1]
Z <- cbind(id, Z)
names(Z)[1] <- "inec_identificador_empresa"
S8va <- Z
# M1[] <- lapply(M, function(x) replace(x, is.infinite(x), NA))
```

O. Eliminación de casos (borrado de celdas) para las validaciones con justificación en los archivos: C2da,...,S2da y C3ra,...,S3ra., etc.

```{r}
reporte_BKUP <- reporte
bases <- c("C2da", "C3ra", "C4ta", "C5ta", "C7ma", "C8va", "D2da", "D3ra", "D4ta", "D5ta", "D6ta", "D7ma", "D8va", "L2da", "L3ra",
"L4ta", "L5ta", "L6ta", "L7ma", "S2da", "S3ra", "S4ta", "S5ta", "S6ta", "S7ma", "S8va")
for (b in bases) {
  comando <- paste0("id_b <- intersect(", b, "$inec_identificador_empresa, reporte$inec_identificador_empresa)")
  eval(parse(text = comando), envir = .GlobalEnv)

  for (fil in 1:length(id_b)) {
    comando <- paste0("var_list_justif <- as.character(apply(", b, "[", fil, ", ], 1, function(x){names(x[which(x==1 | x==2)])}))")
    eval(parse(text = comando), envir = .GlobalEnv)

    var_list_reales <- var_list_justif[which(var_list_justif %in% names(reporte))]

    for (var in var_list_reales) {
      BorrarCeldas("reporte", id_b[fil], var)
    }
    print(paste0("Barridas todas las variables de ", b, "[", fil, ", ]"))
  }
  print(paste0("Completada toda la base ", b))
}
rm(b, bases, id_b, fil, var_list_justif, var_list_reales, var)
rm(comando)
```

P. Insertar la columna de total de errores por empresa, y finalmente exportar a Excel el frame "reporte2".

```{r}
err_x_emp <- apply(reporte[, lista_vars_control], 1, function(x){length(x[which(x > 0.3)])})
reporte2 <- cbind(reporte, err_x_emp)
writexl::write_xlsx(reporte2, "C:/Users/Ramiro/Documents/EMPRESAS 2020/9no Corte - 25-11-2021/Comp_2019_2020_9na.xlsx")
```

ANEXO E. Código R Markdown para la verificación de valores extremos de las variables de escala.

```
---
title: "SX Valores extremos 2020"
author: "Ramiro Benavides"
date: "22/11/2021"
output: html_document
---

1. Carga de la BDD.
```{r}
EMP_2020 <- haven::read_sav("~/Documents/EMPRESAS 2020/9no Corte - 25-11-2021/BDD/Empresasfusionado.sav")

B <- EMP_2020[which(EMP_2020$estado == "C" & EMP_2020$efectividad == 1), ]
# B <- EMP_2020[!is.na(EMP_2020$nombre_critico), ]

# Eliminar primero las empresas no efectivas (dejar únicamente a las efectivas).
B <- B[B$efectividad==1, ]

unique(B$nombre_critico)

# Eliminar a las empresas fusionadas y absorbidas.
B <- B[B$fusionada!="1.7" & B$absorbida!="1.8", ]

# Eliminar a la empresa TAME, que es efectiva y criticada, pero ya no existe.
B <- B[-which(B$inec_identificador_empresa == "41909487178"), ]
```

2. Asignación de crítico por zonal.
```{r}
# Asignar empresas a zonal, según el nombre del crítico.
B$Zonal_DEAGA <- NA_character_
B$Zonal_DEAGA[B$nombre_critico == "MUÑOZ MORA HARLETH ENYTH"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "CASTRO VILLALVA ROBERTO MAURICIO"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "SUMBA AYALA MARCIA IVON"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "AYALA SUMBA  MARCIA IVON"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "TERAN LOJA ADRIAN OSWALDO"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "VERGARA URGILES  REBECA"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "MALIZA CHASI LUZ ANGELICA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "YAUTIBUG BARRERA KATERIN PAOLA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "URGILÉS URGILÉS ENMA CUMANDÁ"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SANCHEZ RAMIREZ GERMAN ANDRES"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "MORETA SARAGURO EVELYN SOFIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "GONZALEZ PIONCE FABRIZIO ADRIAN"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "VALENCIA PEREZ FERNANDO DANIEL"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "BEDOYA PEÑAFIEL ANDRES PATRICIO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "BALSECA VILLALVA LESLIE PRISCILLA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "HOLGUIN QUIJIJE KENYS ELIZABETH"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "NIETO VERA MARYLIN KATHERINE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "VILLACIS VILLACIS VICTOR HUGO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "PEÑA ICAZA ARMANDO BERNARDINO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "BENITES CAMPOVERDE JESSICA DENISE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "RAMOS MUÑOZ JIREMY JACQUELINE"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "ALVARADO CEDEÑO SELENE GARDENIA"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "NACEVILLA CACHAGO MIREYA PATRICIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "GUEVARA ALVARADO ELICEO FRANCISCO"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "GOMEZ MEJIA BYRON DAVID"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "VALDEZ SALAZAR JOSSELIN NOHELY"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SEVILLA VILLACIS CRISTINA SOLANGE"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "HERNANDEZ JATIVA MARIA PAULINA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "PAREDES ROMERO ALEXANDER JAVIER"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "CALIXTO FARIÑO JOHNNY ALEXIS"] <- "Litoral"
B$Zonal_DEAGA[B$nombre_critico == "SEMANATE CAJIAO ANTONIO JAVIER"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "SEMANATE ALVAREZ LEONELA DE LOS ANGELES"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "TOMALA MIRANDA STEFANIE YESENIA"] <- "Litoral"
```
```

```
B$Zonal_DEAGA[B$nombre_critico == "RODRIGUEZ BENAVIDES BRAYAN VLADIMIR"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "RUANO VACA CRISTINA ELIZABETH"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "CALLE PALACIOS MAGALY CARLOTA"] <- "Sur"
B$Zonal_DEAGA[B$nombre_critico == "GUARDERAS NÚÑEZ TATIANA VALERIA"] <- "DICA"
B$Zonal_DEAGA[B$nombre_critico == "PICO SANCHEZ ROSA PATRICIA"] <- "Centro"
B$Zonal_DEAGA[B$nombre_critico == "MAIGUA VELA MAGDALENA DEL CONSUELO"] <- "DICA"
```

3. Determinación de lista de variables numéricas de escala a verificar sus valores extremos.

```{r}
sec_var_sin_trans <- c(7002, 7008, 8007, 8053, 8055, 8098, 8099, 8100, 9001, 9002, 9052, 9058, 9062, 9070, 9094,
                       10000, 10001, 10012, 10020, 10028, 10035)
sec_var_trans <- c(10004, seq(10065, 10527, 21))
lista_vars_sin_trans <- paste0("v", sec_var_sin_trans)
lista_vars_trans <- paste0("t", sec_var_trans)
lista_vars_num <- c(lista_vars_trans, lista_vars_sin_trans)
```

4. Transformación de las variables numéricas de escala que requieren hacerlo.

```{r}
# Creación de variables transformadas para las pertenecientes a la lista "lista_vars_sin_trans".
for (j in 1:length(sec_var_sin_trans)) {
  B[, paste0("t", sec_var_sin_trans[j])] <- B[, paste0("v", sec_var_sin_trans[j])]
  print(paste0("Columna v", sec_var_sin_trans[j], " procesada."))
}

# Creación de variables transformadas para las pertenecientes a la lista "lista_vars_trans".
B$t10004 <- ifelse(B$v10003 == 1, B$v10004 * 2.641722, B$v10004)
for (j in 2:length(sec_var_trans)) {
  for (i in 1:dim(B)[1]) {
    B[i, paste0("t", sec_var_trans[j])] <- ifelse(B[i, paste0("v", sec_var_trans[j]-1)] == 2, B[i, paste0("v", sec_var_trans[j])] * 1000,
ifelse(B[i, paste0("v", sec_var_trans[j]-1)] == 3, B[i, paste0("v", sec_var_trans[j])] * 3.78541, B[i, paste0("v", sec_var_trans[j])]))
  }
  print(paste("Columna", lista_vars_trans[j], "procesada."))
}
```

5. Conversión de 0's en missing y transformación de todas las variables en sus propios logaritmos decimales.

```{r}
l1 <- c(paste0("t", sec_var_sin_trans), paste0("t", sec_var_trans))
for (v in 1:length(l1)) {
  B[l1[v]][B[l1[v]] == 0] <- NA
  B[l1[v]] <- log10(B[l1[v]])
  print(paste("Columna", l1[v], "procesada."))
}
```

6. Función "asignar_grupos()" para segmentar por tamaño el análisis de caja.

```{r}
asignar_grupos <- function (variable) {

tryCatch(rm(caja), error=function(e){}, warning=function(w){})
comando <- paste0("caja <- adjbox(", variable, " ~ B$inec_tamanio, main = 'Caja ajustada [log(", variable, ")]')")
eval(parse(text = paste0(comando)), envir=.GlobalEnv)

tryCatch(rm(vec), error=function(e){}, warning=function(w){})
comando <- paste0("vec <- matrix(nrow=dim(B)[1], ncol=1)")
eval(parse(text = paste0(comando)), envir=.GlobalEnv)

comando <- paste0("grupos <- caja$group[!duplicated(caja$group)]") # Grupos de los extremos hallados (1=Mediana A; 2=Mediana
B; 3=Grande).
eval(parse(text = paste0(comando)), envir=.GlobalEnv)
}
```

7. Función "agregar_vector()" para agregar las nuevas variables de rango a la BDD.

```{r}
```

```
agregar_vector <- function(variable2) {

# Agrega la nueva variable de rango a la BDD y le da un nombre con significado de rango.
comando <- paste0("B['RG", substr(variable2, 4, nchar(variable2)), "'] <- vec")
eval(parse(text = paste0(comando)), envir=.GlobalEnv)
}
```

8. Generación de las variables de reporte de control de extremos "RGxxx" en la base "B".

```{r}
require(robustbase) # Biblioteca que incluye la función adjbox()

for (v in 1:length(l1)) {
  nom_variable <- paste0("B$", l1[v])
  comando <- paste0("condicion <- (dim(B)[1] == as.numeric(summary(", nom_variable, ")[7]))")
  eval(parse(text = comando), envir=.GlobalEnv)
  if (!condicion) {
    asignar_grupos(nom_variable)
    bandera <- paste0("(", nom_variable, " %in% caja$out == TRUE) & B$inec_tamanio == h+2")
    expres_menor <- paste0(nom_variable, "[which(", bandera, ")] < caja$stats[1, h]")
    expres_mayor <- paste0(nom_variable, "[which(", bandera, ")] > caja$stats[5, h]")
    for (h in grupos) {
      if (caja$stats[1, h] < caja$stats[5, h]) {
        comando <- paste0("vec[", bandera, "][", expres_menor, "] <- 'INF'")
        eval(parse(text = comando), envir=.GlobalEnv)
        comando <- paste0("vec[", bandera, "][", expres_mayor, "] <- 'SUP'")
        eval(parse(text = comando), envir=.GlobalEnv)
      }
    }
  } else {
      comando <- paste0("vec <- matrix(nrow=dim(B)[1], ncol=1)")
      eval(parse(text = comando), envir=.GlobalEnv)
  }
  agregar_vector(nom_variable)
  print(paste0("Columna RG",substr(l1[v], 2, nchar(l1[v])), " añadida a la BDD."))
  rm(bandera, caja, comando, condicion, expres_mayor, expres_menor, grupos, h, nom_variable, v, vec)
}
```

9. Generación de la matriz de reporte de valores extremos por empresa.

```{r}
lista_vars_identif <- c("inec_zonal", "Zonal_DEAGA", "inec_tamanio", "inec_ciiu4", "inec_identificador_empresa", "novedad",
                "efectividad", "ruc_muestra", "ruc", "nombre_comercial", "razon_social", "desc_actividad_principal",
                "ciiu4_actividad_principal", "desc_actividad_secundaria", "ciiu4_actividad_secundaria", "nombre_critico",
"fecha_critica")

lista_vars_extremos <- names(B)[grep("^RG", names(B))]

todas_las_vars <- c(lista_vars_identif, lista_vars_extremos)

reporte <- subset(B, select = todas_las_vars)

SUP <- apply(reporte[, lista_vars_extremos], 1, function(x) {length(which(x == 'SUP'))})
INF <- apply(reporte[, lista_vars_extremos], 1, function(x) {length(which(x == 'INF'))})
EXT <- apply(reporte[, lista_vars_extremos], 1, function(x) {length(which(x == 'SUP' | x == 'INF'))})

reporte <- cbind(reporte, SUP, INF, EXT) # Agregar las nuevas columnas de recuento de supremos, ínfimos y extremos al frame
de reporte.
```

10. Exportación a Excel del frame de reporte.

```{r}
writexl::write_xlsx(reporte, "~/Documents/EMPRESAS 2020/9no Corte - 25-11-2021/Reporte_Extremos_2020_28_12_2021.xlsx")
```

Elaborado por	Ramiro Benavides	Firmado electrónicamente por: RAMIRO VICENTE BENAVIDES LEON
Revisado por	Carlos Pilataxi	Firmado electrónicamente por: CARLOS PATRICIO PILATAXI GUAIRACAJA
Aprobado por	Armando Salazar	Firmado electrónicamente por: ARMANDO DAVID SALAZAR MENDEZ

@ecuadorencifras @ecuadorencifras @InecEcuador t.me/ecuadorencifras INEC/Ecuador INECEcuador INEC Ecuador