

Manual de Bases de Datos
Y Análisis de Datos con
Google Data Studio
Comité de Ética de la Investigación

Luis Téllez Ramírez

27 enero 2022

Índice general

1. Conceptos Básicos: Bases de Datos Relacionales	6
1.1. ¿Qué entendemos por datos?¿Qué forma toman los datos?	9
1.2. Bases de datos relacionales	11
1.3. Transformaciones de Datos: Agregaciones	13
2. Tidy Data	16
2.1. Definición	17
2.2. Objetivos	19
2.3. Ejemplos	20
3. Google Data Studio	23

Introducción

“Los datos son el nuevo petróleo.” - Clive Humby, 2006

La primera vez que leí esta frase, vi la importancia que tendrían los datos en el futuro y en el futuro más inmediato, que ya es presente. Pero es cierto que en aquel momento no fui capaz de comprender lo inmensa y profunda que era tal afirmación.

El petróleo a día de hoy es algo fundamental, eso es algo indiscutible, pero no sólo porque lo utilicemos para el transporte, una gran cantidad de plásticos también son derivados del petróleo, asfaltos, lubricantes, componentes de fabricación, generación eléctrica, cableado, aislamientos, ...

La lista es casi interminable, y los usos que aún no se han descubierto de ello. Pero lo cierto es que el petróleo en sí (no soy experto en ello) no se utiliza para prácticamente nada. Aunque parezca una incongruencia con lo que comentábamos previamente, el petróleo en sí mismo, no sirve para nada, son las transformaciones y derivados de éste lo que lo hacen verdaderamente valioso.

Pues lo mismo ocurre con los datos. Los datos en bruto, y muchos (ahora también llamado y mal llamado en muchas ocasiones **Big Data**) a veces no sirven para nada.

Big Data: El Big Data es un concepto relativamente “guay”, que mucha gente cada vez usa más, pero pocos realmente lo utilizan bien. Mucha gente llama “Big Data a aquella hoja de excel que no puede abrir Excel (una hoja de más de 2 gb)”. Podremos hablar de Big Data cuando se reunan los tres siguientes conceptos:

- **Gran cantidad de datos:** Se tenga tal cantidad de datos almacenados, que necesitaremos de bastantes servidores para poder almacenar dicha cantidad.
- **Gran Heterogeneidad de los datos:** Se almacenará todo tipo de datos, desde imágenes, textos, series temporales, datos estructurados, semiestructurado, ... Aquí jugará un papel fundamental lo que denominamos **ETL**, que se verá más adelante.
- **Gran cantidad de datos en Streaming:** No solo tendremos una gigantesca cantidad de datos almacenados, sino que además, no paramos de recibir nuevos datos de manera continua.

Si se reunen esos tres conceptos, yo considero que se utiliza Big Data. Para ello se ha utilizado la definición aportada en este libro del CSIC.

El Big Data supone un mundo de oportunidades, pero a la vez, otro de problemas, ya que la estadística tradicional no sirve para la cantidad de datos en Streaming, y los procesos de Optimización juegan un papel vital. Ya que si queremos sacar conclusiones de los datos en un momento determinado, los necesitamos para ahora, no para dentro de días.

Surgen nuevas metodologías para abordar este tipo de problemas, sobre todo el uso de servidores u ordenadores trabajando en paralelo usando tecnologías del tipo **Hadoop, HIVE, Parquet, APACHE Spark...**

A lo largo del capítulo 1, se dará una visión acerca de las distintas estructuras de datos que se utilizan, pero pondremos el foco en Tidy Data y los Datos estructurados. Para posteriormente, poder introducir una herramienta de Analítica de Datos muy potente, pero que requerirá del uso de Datos estructurados para poder trabajar, de ahí la importancia de manejar los datos de una manera determinada.

Tidy Data y Datos Estructurados

En esta breve sección, quiero plantear el objetivo de este documento para que el lector se sitúe correctamente, sepa de dónde partimos y hacia dónde queremos llegar.

En los párrafos anteriores hablamos de Big Data, una masa ingente de datos de distinto tipo sin ningún orden. Lo normal, es que nosotros jamás nos acerquemos a un problema de estas características, y lo que busquemos sea todo lo contrario; tomar datos atendiendo a una estructura, jerarquía, disponibilidad, seguridad, orden. Conceptos que veremos más adelante.

A lo largo de los capítulos 1 y 2, se presenta el concepto **Base de Datos, datos tabulares** y **Tidy Data**. Buscamos este último, porque se ha demostrado en diversos artículos que es el formato óptimo para los procesos posteriores.

La idea es que si somos conscientes a priori de la importancia que tendrán los datos en el proceso, nos molestaremos en pensar acerca de la estructura óptima que deberá seguir la obtención de datos para facilitar y garantizar que el proceso posterior es sostenible y eficaz.

Primero, utilizamos la siguiente imagen para situarnos correctamente:

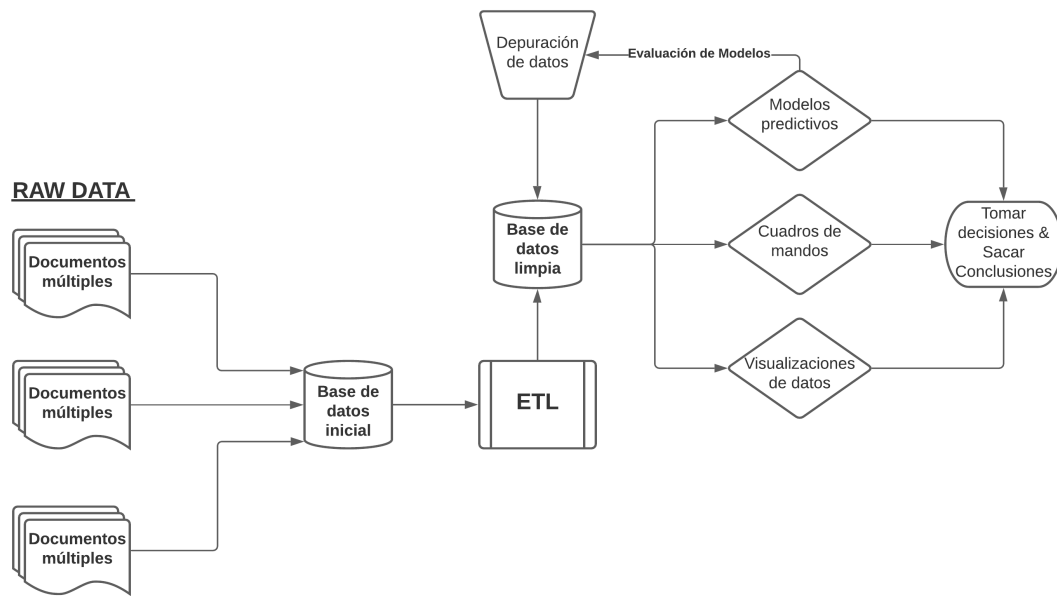


Figura 1: Proceso completo de Datos.

Cuanto más se parezca la **base de datos inicial** a la **base de datos limpia**, mayor consistencia tendrá el proceso.

Por eso es importante entender qué son las bases de datos estructuradas. Una vez llegado ese punto, veremos qué es tidy data, que no deja de ser una transformación más de los datos, para conseguir una calidad superior, aunque evidentemente, cuando se están recogiendo datos, lo normal es que no se tomen en un formato tidy, ni siquiera estructurado. Pero si se sabe que esos formatos son óptimos, nos molestaremos en mejorar al principio, para luego ahorrar y poder sacar mejor conclusiones en el futuro.

Capítulo 1

Conceptos Básicos: Bases de Datos Relacionales

Empezaremos proponiendo algunas definiciones a modo de consenso:

- **Tubería o Canalización de Datos:** En cualquier equipo de ingeniería de software, una canalización es un conjunto de procesos automatizados que permiten a los desarrolladores y a los profesionales de DevOps compilar, construir y desplegar su código en sus plataformas informáticas de producción de forma fiable y eficiente.
- **Ingeniería de Datos:** Los ingenieros de datos son los diseñadores, constructores y gerentes de las tuberías de datos. **Desarrollan la arquitectura, los procesos y son dueños del rendimiento y la calidad de los datos de la solución general.** Para ello, deben ser especialistas en la arquitectura de sistemas distribuidos y la creación de tuberías confiables, combinando fuentes de datos y construyendo y manteniendo almacenes de datos.

Como se puede apreciar, el Ingeniero de Datos va a ser el que plantea la solución acerca de cómo estructurar las bases de datos y organizar las ETL's. Será vital en el proceso, ya que la calidad del dato vendrá en gran parte determinada por la calidad de la Ingeniería de Datos aplicada.

- **ETL:** (Extract - Transform - Load) ETL es un tipo de integración de datos que hace referencia a los tres pasos (extraer, transformar, cargar) que se utilizan para mezclar datos de múltiples fuentes. Se utiliza a menudo para construir un almacén de datos. Durante este proceso, los datos se toman (extraen) de un sistema de origen, se convierten (transforman) en un formato que se puede almacenar y se almacenan (cargan) en un data warehouse u otro sistema. Extraer, cargar, transformar (ELT) es un enfoque alternativo pero relacionado diseñado para canalizar el procesamiento a la base de datos para mejorar el desempeño.

- **Ciencia de Datos:** Es algo difícil de definir, pero una buena manera de entender qué es y a la vez cómo funciona, es ver esta rama como la intersección de tres ramas. Estas ramas son:
 - **Matemáticas y Estadística:** Jugarán un papel vital en la credibilidad de las decisiones aportadas una vez extraída toda la información de los datos.
 - **Informática:** Será importante tener un buen conocimiento sobre funcionamiento de servidores, redes, desarrollo de software, programación...
 - **Business Intelligence:** También conocido como inteligencia de negocio, en nuestro caso como científicos de datos, debemos ser capaces de mezclarnos con gente que es experta en el tema sobre el que vamos a aportar una solución para ser capaces de entender el contexto sobre el que se está desarrollando el problema.

En definitiva, es la siguiente intersección:

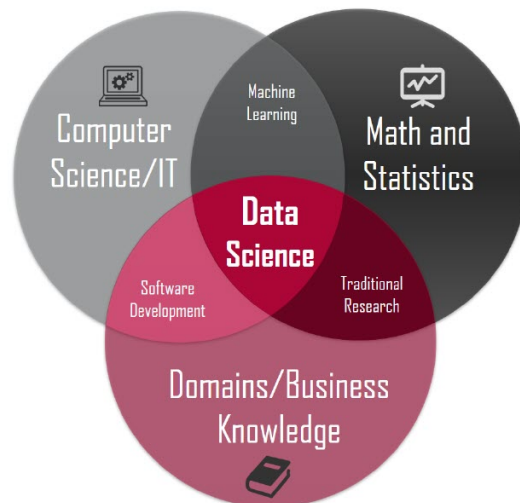


Figura 1.1: Data Science

¿Cuál es la función de un Científico de datos? Crear productos en torno a los datos.

La Ciencia de Datos es un proceso de *principio a fin*:

- **Orientado a producto/cliente:** validado en escenarios reales.
- **Iterativo:** Experimental y orgánico, con mejoras y adaptaciones continuas.
- **Data Driven:** Basado en el estado de los datos.

No olvide que es una **Ciencia**, es decir, sigue el **Método Científico**:

- Preguntas

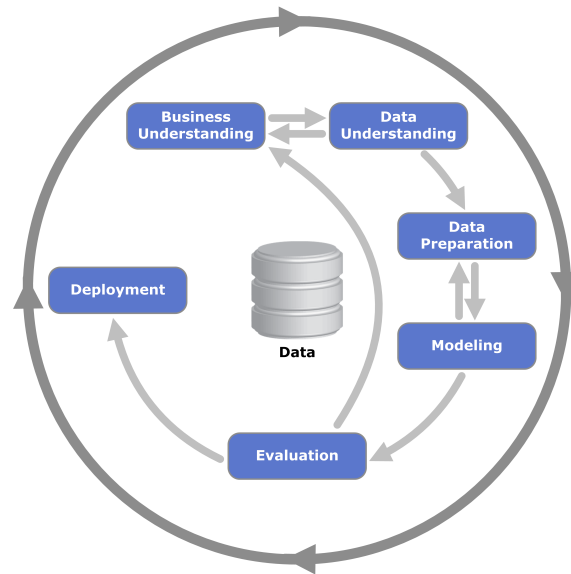


Figura 1.2: CRISP-DM

- Investigación
- Hipótesis
- Experimentos
- Conclusiones \Rightarrow Fracaso \Rightarrow Preguntas \Rightarrow Investigación ...
- Comunicación

El ciclo de vida de los datos:

- Extracción
- Preprocesamiento
- Almacenamiento
- Exploración
- Modelado
- Visualización

Al final, aproximadamente el 80 % de lo que consideramos como análisis de datos no lo es realmente. La mayor parte del tiempo se dedica a transformaciones y traslado de datos. Un formato o sistema nunca es óptimo para todos los procesos, los cuales requieren distintos perfiles en cada etapa.

Todo esto es importante de entender, ya que, al igual que un chef profesional no podría preparar un buen plato, un científico de datos no podrá desarrollar buenos productos si los datos no son buenos.

Por eso es claro que los buenos datos construirán mejores modelos.

Tras esta pequeña descripción de lo que sería la Ciencia de Datos, procedemos con la exposición del capítulo.

- **Análisis de Datos:** Entenderemos por *análisis de datos*, una rama de la Ciencia de Datos, que se basa en la toma de decisiones mediante el uso de visualizaciones, modelos desarrollados en otras ramas de la Ciencia de Datos, estadísticas, ... etc.

1.1. ¿Qué entendemos por datos? ¿Qué forma toman los datos?

Lo cierto es que para los humanos, el formato tabular resulta natural, pero no todos los datos son estructurados, existen distintos tipos de datos.



Figura 1.3: Dato no estructurado, semiestructurado, estructurado

- **Dato no estructurado** puede ser un número indeterminado de ficheros excel con una o varias hojas en su interior, documentos de texto sin estructura, corporativos, logs, multimedia, señales, ficheros binarios o con formato privativo, mediciones de sensores o equipo industrial, imágenes ...
- **Los datos semiestructurados** proporcionan una mayor adaptabilidad a diversas situaciones, es un tipo de dato menos estándar, que requiere una mayor especialización tecnológica, pero que también puede ser muy potente. Algunos tipos son JSON, YAML, XML, NOSQL, GRAFOS, HTML...
- **Los datos estructurados** son modelos estrictos y adaptados a procesos particulares. Esto supone un pequeño problema si queremos estar permanentemente cambiando la manera en la que recogemos los datos por ejemplo. Sin embargo, es un tipo de dato bien estandarizado a lo largo de los años (SQL y RDBMS). Son demasiado rígidos y costosos en ciertos aspectos, y pueden presentar problemas de falta de adaptación y escalabilidad.

Según el IDG Research 2018, 1 de cada 3 proyectos relacionados con los datos fallan, y la causa principal de estos errores es la gestión de los datos.

Los datos deben ser:

- Compartidos
 - Accesibles
 - Centralizados
 - Gobernados
 - Securizados
 - Compliant
- Consultables
 - Catalogados
 - Indexados
 - Estables
 - Adaptables
- Compatibles
 - Completos
 - Limpios
 - Confiables
 - Normalizados

1.2. Bases de datos relacionales

Nuestro objetivo será llegar finalmente a Tidy Data, entonces vamos a definir y entender lo que son las bases de datos relacionales sin verlo demasiado profundo, ya que es algo bastante complejo y se deja también como actividad para el lector.

Los datos tabulares deben interpretarse de la siguiente manera:

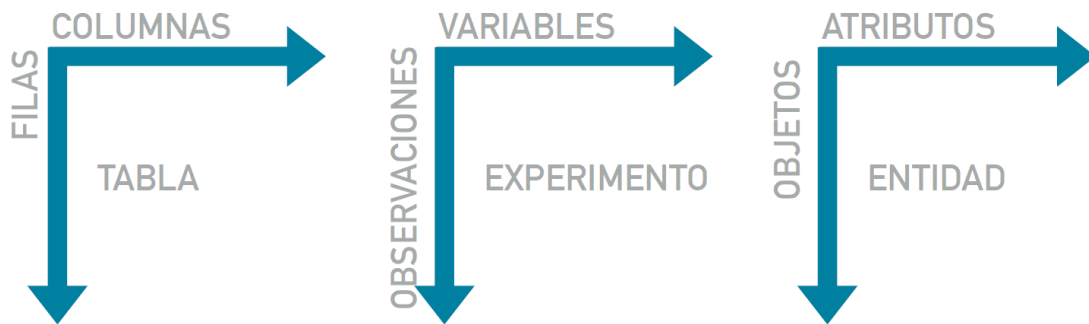


Figura 1.4: Interpretación datos tabulares

El modelo de bases de datos relacionales fue formulado en 1970 por Edgar Frank Ted Codd.

El modelado de bases de datos no es una tarea sencilla, donde debe apoyarse de expertos en la materia para plantear este tipo de soluciones.

Las bases de datos relacionales están basadas en definir **entidades** y **atributos**. Podemos entender esto como *definir el molde sobre el que haremos las galletas*.

Otra cosa muy importante es el **tipado de datos**. No solo hay que tomar los datos, sino que hay que tomarlos de la mejor manera posible. Teniendo en cuenta que, por ejemplo, la **Edad** es una variable numérica, incluso se podrían poner validaciones, del tipo, que la Edad sea un número entero (normalmente entre 0 y 100).

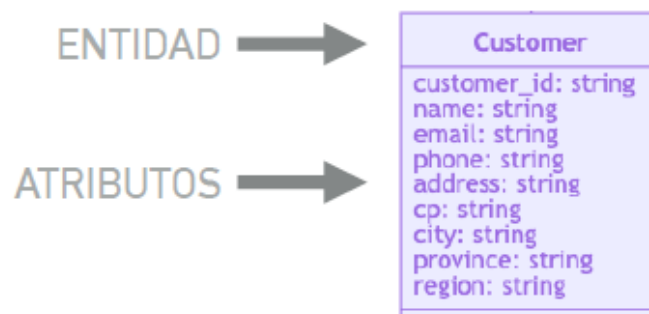


Figura 1.5: Entidad

La entidad es “Customer” (Cliente), imaginemos que queremos modelar un cliente. Pensamos qué caracteriza a un cliente.

- Customer_id: Id incremental que se añade por defecto en una tabla de la base de datos.
- Nombre
- Email
- Teléfono
- Dirección
- Código Postal
- Ciudad
- Provincia
- Region
- ...

Pero claro, si se hace de esta manera estaríamos creando una única tabla (Cliente) que tiene todos esos atributos correspondientes al cliente. Bien, pero qué ocurre ahora, el cliente es único, es decir, el cliente XXXX con DNI XXXXXXXXXX es irrepitible, sin embargo, en varios pedidos que me ha hecho, me ha aportado varias direcciones.

Nos enfrentamos entonces al primer problema. Ya que tal y como está planteado, tendríamos que crear un cliente (una entrada en la BD) y para poner varias direcciones, duplicar dicha entrada cambiando una parte. Incluso si queremos poner la misma dirección, tenemos problemas de duplicidad.

ID_CLIENTE	NOMBRE	EMAIL	CIUDAD	DIRECCION
001	Jafet Calderón-Acosta	pacosainz@example.net	Cáceres	Alameda de Chita Martinez 9 Apt. 78 \nValladol...
002	Jafet Calderón-Acosta	maximilianogabaldon@example.net	Toledo	Cañada de Yéssica Diez 76\nCórdoba, 05254

Figura 1.6: Cardinalidad

Esto es esencialmente erróneo, ya que, podríamos crear dos tablas separadas que tuviesen la siguiente relación (cardinalidad).

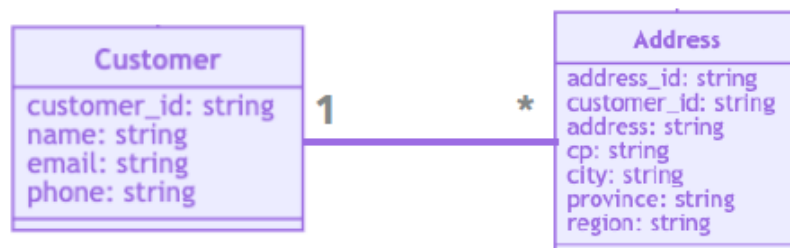


Figura 1.7: Cardinalidad

Así, se pueden hacer dos tablas, una **Cliente** y otra **Direcciones**, relacionadas y con *cada cliente pudiendo tener más de una dirección*.

	customer_id	name	email	phone
0	customer_0000	Alberto Catalán Zaragoza	ismaelfabregat@goni.com	+34 993 847 912
1	customer_0001	Alicia Comas Mosquera	dzaenrique@zabaleta.info	+34368 421 135

	address_id	customer_id	address	cp	city	province	region
0	address_0000	customer_0000	Callejón de Jose Antonio Valls 421	02271	Ferrol	La Coruña	Galicia
1	address_0001	customer_0001	Urbanización Gonzalo Cerdán 56 Apt. 57	67871	Motril	Granada	Andalucía
2	address_0002	customer_0001	Avenida Hugo Galán 48	05274	Badajona	Barcelona	Cataluña
3	address_0003	customer_0001	Glorieta de Antonio Olivares 6 Apt. 83	67572	Torrelavega	Cantabria	Cantabria

Figura 1.8: Ejemplo Cardinalidad

Además, existe un lenguaje específico, el SQL (Structured Query Language) que nos permitirá hacer consultas sobre estas tablas.

Hemos visto ejemplos muy sencillos, pero esto podría complicarse todo lo que se quisiera, ya que, para un ERP (Enterprise Resource Planning - Sistema de planificación de recursos empresariales), podríamos pensar en un cliente que tiene varias direcciones y asociada a cada dirección tiene varios pedidos, cada pedido con distintos productos y a su vez el mismo producto puede tener distintas marcas ...

1.3. Transformaciones de Datos: Agregaciones

Imaginemos la siguiente situación:

Tenemos los datos de distintas compras que se han efectuado a distintos comercios por distintas personas:

	nombre	direccion	email	ciudad	fecha_compra	Supermercado	Gasto Compra
0	Luisa Arce-Fábregas	Ronda de Fausto Cadenas 48\nÁlava, 45486	patriciomontesinos@example.com	Alicante	2020-09-24	Mercadona	18
1	Cecilio Peral-Saez	Rambla Régulo Montes 33\nPontevedra, 18491	ricarda14@example.org	Guadalajara	1978-11-03	Dia	101
2	Martina Cortés	Cañada Edelmira Ros 50\nPalencia, 32178	teofilafuertes@example.com	Toledo	2018-12-26	Mercadona	34
3	Susana Manrique	Plaza Josefa Tapia 40\nGranada, 82715	noelia78@example.com	Girona	2020-05-07	Lidl	292
4	Carmela Bermejo-Rosell	Ronda Susanita Belda 87 Apt. 25 \nJaén, 29820	psamper@example.com	Murcia	2003-05-20	Carrefour	325
...
995	Nélida Jacinta Cantón Uribe	Camino Fidel Serrano 42\nZamora, 76267	ibarco@example.net	Segovia	1980-03-18	Dia	177
996	Natanael Amigó Vera	Acceso Elba Cervantes 52 Apt. 73 \nSegovia, 70531	anguitabenigna@example.org	Soria	1997-09-06	Mercadona	284
997	Anastasia Riera Tello	Paseo de Elpidio Palacio 167 Apt. 49 \nGranada...	victor29@example.com	Sevilla	1987-07-22	Carrefour	332
998	Nieves Calvo Marquez	Calle de Jose Francisco Gimenez 46\nCiudad, 31204	mayte27@example.net	Zaragoza	2004-12-19	Lidl	336
999	Martin Ruiz Hidalgo	Plaza de Elena Herranz 4\nLleida, 49502	marino32@example.net	Teruel	2015-01-04	Carrefour	83

1000 rows × 7 columns

Figura 1.9: Ejemplo Compras

Tenemos los datos en formato tabular, y por ahora está desagregado, pero por ejemplo, podríamos preguntarnos lo siguiente:

- ¿Cuánto ha recaudado cada Supermercado.
- ¿Qué ciudad es la que más dinero ha gastado. Cuánto se gasta por ciudad.
- ¿Cuál ha sido el año en el que más se ha gastado. Cuánto se gasta anualmente.

Algunos ejemplos de agregaciones:

Gasto Compra	
fecha_compra	
2022	130
2021	3535
2020	3207
2019	5364
2018	4410
2017	4405
2016	2790
2015	1861
2014	3244
2013	4650
2012	3267

Figura 1.10: Agregación por año

Gasto Compra		
ciudad	Supermercado	
Ávila	Mercadona	1701
	Lidl	902
	Dia	772
	Carrefour	1410
Álava	Mercadona	648
...
Alicante	Mercadona	969
Albacete	Lidl	1575
	Dia	242
	Mercadona	942
	Carrefour	1013

Figura 1.11: Agregación por supermercado y ciudad

[53]:

Gasto Compra	
Supermercado	
Lidl	46742
Mercadona	46677
Dia	46650
Carrefour	45367

Figura 1.12: Agregación por supermercado

Pero estos nuevos datos que estamos viendo, también tienen forma de tabla. Bien, aquí es donde encontramos el gran problema y es por el que estamos realizando esta especie de “manual” o “guía”. Aunque las agregaciones tengan un formato tabular, son datos que están “**pivotados**”, y aunque nos aporten **información**, ya han dejado de servir para analítica. Es decir, aporta información por un lado, pero se pierde mucha por otra.

Para nosotros poder visualizar la información que estamos obteniendo aquí en cada agregación que hacemos, lo que haremos será lo siguiente:

1. Nos aseguraremos que los datos que subamos a Google Data Studio sean datos tabulares, en forma desagregada.
2. Desarrollaremos nuestro cuadro de mandos y haremos las transformaciones pertinentes (agregaciones) para responder a cada una de las preguntas que nos puedan surgir.

Pero lo más importante, y se debe hacer hincapié en ello, es que no se suban datos que ya han sido previamente agregados a Google Data Studio. Ya que después tendremos muchos problemas a la hora de seguir trabajando con ellos.

Para entender la manera correcta en la que se deben subir los datos, estudiaremos una estructura de datos que fue definida por Hadley Wickham como Tidy Data.

Capítulo 2

Tidy Data

El capítulo 2 se llama **Tidy Data** porque es la estructura propuesta por el estadístico y desarrollador de Software para R, Hadley Wickham, propuesto en el Journal of Statistical Software, el 1 de enero de 2014.

A lo largo de este capítulo, se señala y estudia por qué es una estructura de datos muy eficiente para todo el desarrollo posterior, es decir, analítica de datos, ciencia de datos, estadística ...

Se ha visto visto que un formato tabular o relacional es quizás el más organizado o rico en detalles para almacenar nuestros datos. No obstante este formato no garantiza una representación óptima de los datos de cara a minimizar espacio, flexibilidad y facilidad de manipulación.

En un proceso de análisis de datos dedicamos en torno al 80 % del tiempo a **limpiar** los datos (**Data Cleaning**), y aunque en la mayoría de los casos esto incluye una gran cantidad de trabajo transformando nuestros datos o extrayendo información a partir de formatos no estructurados, también se realizan muchas tareas procesando y limpiando datos en formatos estructurados.

Es posible encontrar duplicidades en los datos, podemos encontrar valores faltantes, inconsistencias en los datos, datos anómalos, ...

El trabajo de un analista o de un científico de datos es el análisis de los datos después de haberse recopilado, por lo que debemos adaptar su formato para que sea lo más estándar posible y compatible con las herramientas a utilizar. *Puede que el formato original de los datos* sea más adecuado para su recopilación o directamente sea erróneo.

Tidy data es un esfuerzo por crear un estándar en el formato de los datos estructurados de cara a desarrollar **técnicas y tecnologías** estandarizadas a su vez para manipulación y análisis de datos. Dada la enorme cantidad de herramientas disponibles en la comunidad muchas veces acabamos dedicando demasiado tiempo a adaptar la salida de una tecnología concreta para poder utilizarla como entrada de otra. Unas pautas de modelado estandarizadas permitirán que la interfaz de nuevas

herramientas se unifique.

Además el modelo relacional normalizado propuesto por Codd está planteado en un lenguaje alejado del lenguaje **estadístico**, Tidy Data puede verse como una reinterpretación del modelo relacional en términos naturales y más útiles para el análisis de datos y la estadística. Esto implica también que algunas de las pautas que impone el estándar ayuden también a optimizar la representación de la información no solo a nivel instrumental sino a nivel estadístico.

El estándar Tidy data fue propuesto por Hadley Wickham, un estadístico que ha trabajado tanto en el campo académico como en el empresarial (Director científico en Rstudio). El trabajo original contextualiza el estándar dentro de las herramientas del lenguaje de programación R, no obstante los principios básicos se explican de manera genérica para cualquier conjunto de datos, lo que ha permitido extrapolar sus resultados al conjunto de herramientas para la ciencia de Datos.

El paper puede encontrarse públicamente y es una lectura básica para cualquier profesional de este campo.

2.1. Definición

El estándar tidy data se basa en tres reglas sencillas:

1. Cada variable es representada por una columna.
2. Cada observación representada por una fila.
3. Cada unidad observacional es representada por una tabla.

Cualquier dataset que no este organizado de esta manera se denomina messy data o datos desorganizados.

Ejemplo

El siguiente ejemplo muestra un como estructurar un mismo problema de distintas formas, una de ellas tidy.

El problema consta de tres variables:

- “patient” con tres posibles valores (John, Mary, Jane)
- “treatment” con dos posibles valores
- “result” con 5 valores más la posibilidad de estar perdido (-, 16, 3, 2, 11, 1)

Se han realizado una experimentación cruzada en la que se le ha aplicado un tratamiento a cada paciente y se ha medido el resultado. Los datos podrían haberse recopilado de la siguiente manera durante el estudio original:

	treatmenta	treatmentb
John	-	2
Jane	16	11
Mary	3	1

En otro contexto los datos podrían haberse recopilado por tratamiento:

	John	Jane	Mary
treatmenta	-	16	3
treatmentb	2	11	1

Estos formatos no nos permiten identificar los elementos experimentales, no obstante si nos ceñimos al estándar tidy el resultado es mucho más representativo.

name	treatment	result
John	a	-
Jane	a	16
Mary	a	3
John	b	2
Jane	b	11
Mary	b	31

El formato tidy nos permite determinar claramente que columnas son observaciones y cuales variables, relaciones funcionales entre ellas, restricciones y una única forma de extraer toda la información del dataset. Además permite identificar mejor cualquier anomalía, como por ejemplo los casos perdidos.

Como ya podemos anticipar, transformar los datos que podamos tener al formato “Tidy Data” va a ser el objetivo principal.

Sin embargo, no hay una solución única a cada problema ni una receta básica que podamos aplicar, pero si podemos identificar patrones comunes en cada dataset. Como cita Wickham en el paper original:

“Happy families are all alike; every unhappy family is unhappy in its own way”. - Leo Tolstoy

Lo que viene a decirnos, que los buenos datasets son más o menos todos iguales y no suelen requerir demasiadas transformaciones, pero en datos que están mal recogidos, podríamos encontrar cualquier cosa.

2.2. Objetivos

Obtener el formato Tidy de los datos

Todos los datasets originales (raw) se encuentran en formatos no ordenados y es necesario transformarlos para poder generar su version tidy.

Utilización de técnicas reproducibles y entender las transformaciones de manera “agnóstica” al lenguaje

En el paper original de tidy data se definen las operaciones de manera general, explicado las transformaciones sobre los datos de manera independiente o agnóstica a cualquier lenguaje antes de aplicar instrucciones propias de R. Es muy importante no obsesionarse con los detalles instrumentales de herramientas particulares como “pandas” y quedarse con las estrategias y técnicas a nivel teórico.

Poder realizar estas transformaciones puede no ser algo tan “sencillo” como a primera vista podría pensarse, sobre todo si los datos son masivos. En casos sencillos podría hacerse a mano si son pocos datos, teniendo un buen manejo de herramientas como Excel.

Intentaremos utilizar funciones de alto nivel y escribir un código lo más reproducible posible, cercano casi al lenguaje natural, que permita entender y reproducir el proceso de transformación con solo leerlo.

Un buen recurso para ir absorbiendo el vocabulario es la siguiente referencia de **pandas** enfocada más al lenguaje de la Ciencia de Datos que a la propia librería.

- Python Cheat sheet
- Pandas Cheat Sheet
- R data Wrangling Cheat sheet

Limpieza adicional de los datos

Además de obtener el formato tidy, vamos a realizar una serie de transformaciones sobre los datos que limpien aun más el formato y obtengan un dataset cómodo y representativo de cara a hacer exploración de datos y modelado.

1. El nombre de las variables cumplirá las restricciones de python en cuanto caracteres, empezará en minúscula y utilizara “snake_case”.
2. Las variables estarán ordenadas de manera lógica a la unidad observacional representada. En primer lugar colocaremos variables experimentales y en segundo lugar observaciones, además intentaremos preservar órdenes jerárquicos y colocaremos factores o categorías antes de variables numéricas.
3. Los casos del dataset se ordenarán conforme a patrones temporales o alfabéticos en función de sus variables.

2.3. Ejemplos

En primer lugar, veamos uno de los ejemplos más comunes que podríamos encontrar:

Problema 1: Cabeceras de columnas como valores

Este es uno de los problemas más comunes en datasets desorganizados. Si bien es un formato comprimido que puede resultar útil para su almacenamiento o introducción manual, resulta inconveniente para su análisis ya que no todos los datos están almacenados como valores.

El dataset de ejemplo “pew” explora la relación entre los ingresos y la religión de diversos ciudadanos americanos consultados por un centro de investigación.

Se subirán los datos para que se tengan disponibles y pueda seguirse el ejemplo:

```
df_pew = pd.read_csv("./data/pew-raw.csv")
df_pew
```

Out[3]:

	religion	<\$10k	\$10-20k	\$20-30k	\$30-40k	\$40-50k	\$50-75k
0	Agnostic	27	34	60	81	76	137
1	Atheist	12	27	37	52	35	70
2	Buddhist	27	21	30	34	33	58
3	Catholic	418	617	732	670	638	1116
4	Dont know/refused	15	14	15	11	10	35
5	Evangelical Prot	575	869	1064	982	881	1486
6	Hindu	1	9	7	9	11	34
7	Historically Black Prot	228	244	236	238	197	223
8	Jehovahs Witness	20	27	24	24	21	30
9	Jewish	19	19	25	25	30	95

Figura 2.1: Visualización dataframe *pew*.

Nos encontramos frente a una tabla pivotada, dónde realmente, estamos visualizando **3 variables**, que son:

- Religion.
- Ingresos.
- Frecuencia absoluta de los valores.

Bien, ahora, tenemos que hacer las transformaciones pertinentes para llegar al mismo dataframe pero en formato tidy:

Out[3]:

	religion	income	freq
0	Agnostic	<\$10k	27
1	Agnostic	\$10-20k	34
2	Agnostic	\$20-30k	60
3	Agnostic	\$30-40k	81
4	Agnostic	\$40-50k	76
5	Agnostic	\$50-75k	137
6	Atheist	<\$10k	12
7	Atheist	\$10-20k	27
8	Atheist	\$20-30k	37
9	Atheist	\$30-40k	52
10	Atheist	\$40-50k	35
11	Atheist	\$50-75k	70

Figura 2.2: Visualización dataframe *pew* en formato tidy.

Para obtener la versión tidy de este dataset necesitamos realizar las siguientes operaciones:

- Una operación de “reshape” para transformar las columnas que representan un valor (colvars) a una sola columna, mapeando el dato correspondiente de la tabla y manteniendo el valor de la religión fijo para dicho caso. En pandas esta operación se denomina “melt”.
- En este caso el dataset se ha ordenado por la variable “religion” e “income”, que deben aparecer en ese orden por ser el campo de agrupación.
- En el caso de la variable income lo ideal es reordenar las categorías por orden ascendiente, para ello es necesario transformar la columna de un valor “object”

que representa un “string” a un valor “category” y establecer un orden entre las categorías.

```
income_categories = df_pew.columns[1:].tolist()

df_pew_tidy_ex = (
    df_pew
    .melt(id_vars=['religion'], value_vars=
          income_categories, var_name="income", value_name="
          freq")
    .assign(
        income=lambda df: pd.Categorical(df.income).
            reorder_categories(income_categories)
    )
    .sort_values(["religion", "income"])
)
```

Y con las transformaciones que acabamos de hacer, ya tenemos lo siguiente:

	religion	income	freq
0	Agnostic	<\$10k	27
10	Agnostic	\$10-20k	34
20	Agnostic	\$20-30k	60
30	Agnostic	\$30-40k	81
40	Agnostic	\$40-50k	76
50	Agnostic	\$50-75k	137
1	Atheist	<\$10k	12
11	Atheist	\$10-20k	27
21	Atheist	\$20-30k	37
31	Atheist	\$30-40k	52
41	Atheist	\$40-50k	35
51	Atheist	\$50-75k	70
2	Buddhist	<\$10k	27
12	Buddhist	\$10-20k	21
22	Buddhist	\$20-30k	30
32	Buddhist	\$30-40k	34

Figura 2.3: Visualización dataframe *pew* en formato tidy.

Tener los datos en este formato es muy cómodo, ya que nos permitirá exportarlo a una excel o csv, y de ahí, llevárnoslo a herramientas como Google Data Studio, donde podremos trabajar en el desarrollo de cuadros de mando interactivo para explotar estos datos de manera eficiente.

Capítulo 3


Google Data Studio

Con un comando similar al siguiente, podemos exportar los datos a un archivo excel. Este archivo excel, nos los llevaremos a Google Drive

```
df_pew_tidy_ex.to_excel('archivo_datos_GoogleDS.xlsx')
```

Ahora, es importante que sigamos los siguientes pasos de cara a empezar a usar Google Data Studio.

1. Creamos una carpeta en Google Drive, en mi caso, MANUAL_GOOGLE_DS, y subimos el archivo **Excel** con el que estábamos haciendo las transformaciones. Debemos tener lo siguiente:



The screenshot shows the Google Drive interface. At the top, there is a breadcrumb navigation: 'Mi unidad > MANUAL_GOOGLE_DS'. Below this is a table listing the contents of the folder. The table has four columns: 'Nombre' (Name), 'Propietario' (Owner), 'Última modificaci...' (Last modified), and 'Tamaño de archivo' (File size). There is one file listed: 'archivo_datos_GoogleDS.xlsx' with a green 'X' icon, owned by 'yo' (me), last modified at '11:56', and a size of '6 kB'.

Nombre ↓	Propietario	Última modificaci...	Tamaño de archivo
 archivo_datos_GoogleDS.xlsx	yo	11:56	6 kB

Figura 3.1: Subida archivo.

2. Una vez hemos subido el archivo, lo abrimos como Excel desde Google, y en **Archivo**, procedemos a: “Guardar como una hoja de cálculo de Google”. Ver imagen 3.2.
3. Una vez hecho este paso, volvemos a la carpeta original de Google Drive, donde ahora, en vez del archivo Excel, debemos tener dos archivos; La Excel por un lado, y el archivo de GoogleSheets. Ver imagen 3.8.

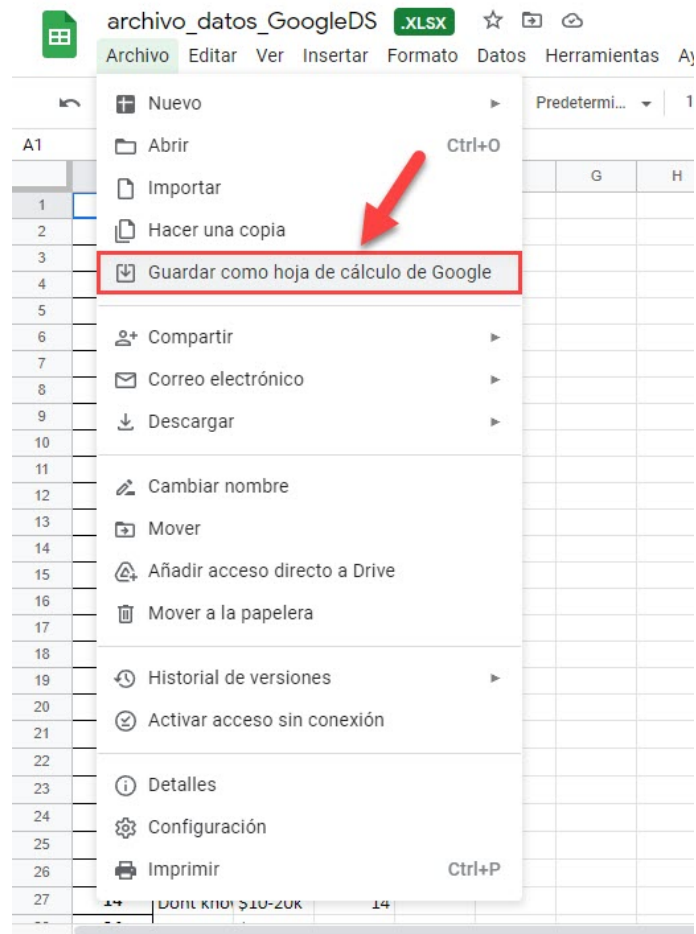


Figura 3.2: Convertir archivo.

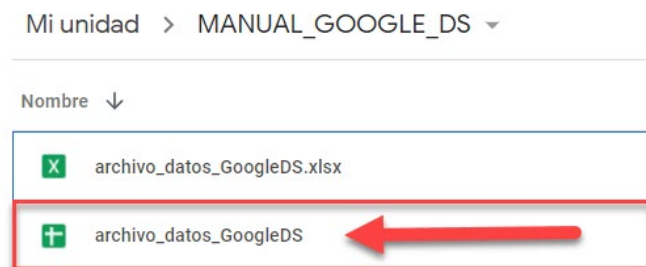


Figura 3.3: Archivos Google Drive.

Ahora, abrimos Google Data Studio y nos dirigimos a la sección **Fuentes de Datos**.

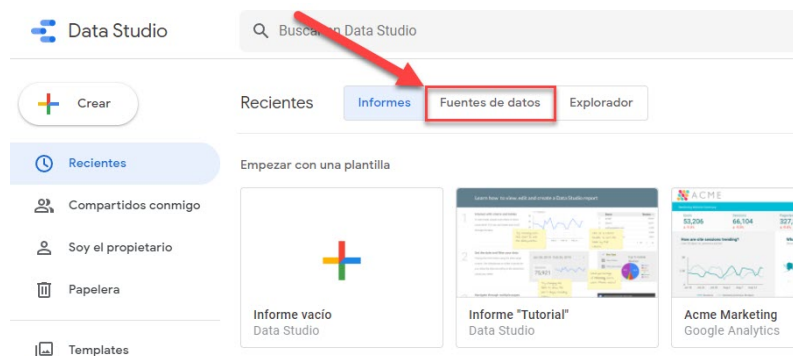


Figura 3.4: Fuentes de Datos.

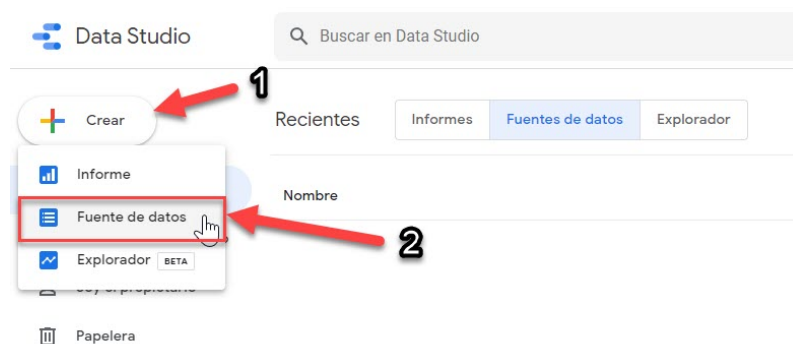


Figura 3.5: Creamos una fuente de Datos.

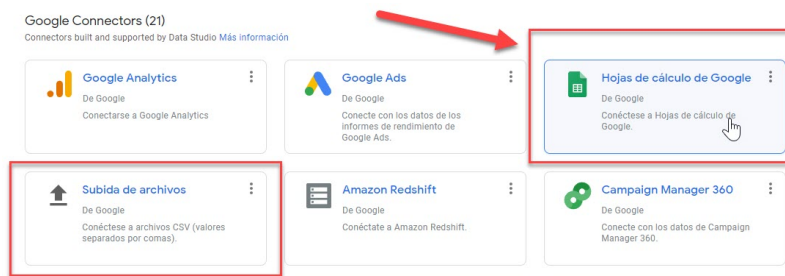


Figura 3.6: Selección de conector.

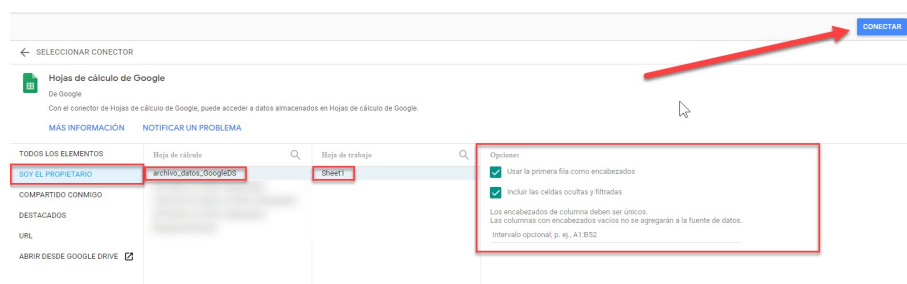


Figura 3.7: Conectamos la base de datos.

Ahora, se revisa la base de datos para confirmar que Google Data Studio ha interpretado correctamente el tipado de los datos.

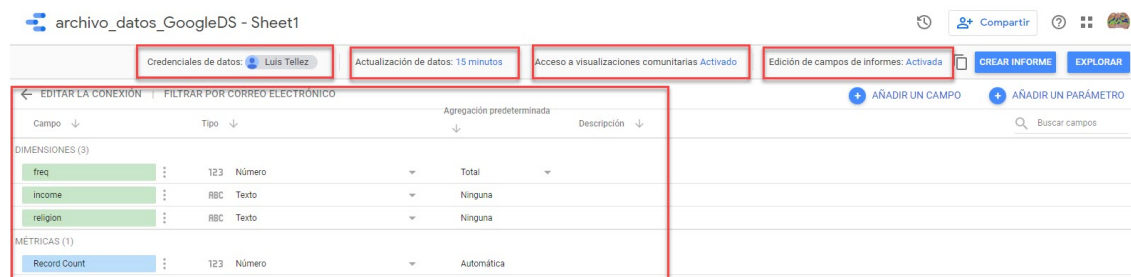


Figura 3.8: Revisamos la base de datos.

En las opciones de arriba, podemos ver lo siguiente:

- Credenciales del propietario.
- Actualización de datos. Cada 15 minutos, esto quiere decir, que si estamos cambiando los datos en streaming, el cuadro de mandos que tengamos desplegado se actualizará cada 15 minutos atendiendo a los nuevos datos.

No tocamos nada más y pulsamos en **CREAR INFORME**.

Ahora, es difícil continuar el manual por aquí, y por tanto lo dejaremos aquí.

Para continuar, recomiendo seguir algún tutorial de Google Data Studio (hay muchos, incluso cursos en Udemy y demás).

Una vez se tiene en cuenta todo lo expuesto en este documento, proceder a la creación de un informe en Google Data Studio es lo “más sencillo”, pero no por ello lo menos importante.

Dentro de ser el paso más sencillo, puede ser bastante complejo sacar el jugo de los datos, y depende en gran medida de la creatividad que se tenga, los conocimientos sobre el manejo de datos ... etc.

Os dejo algún tutorial de Google Data Studio.

Tutorial Google Data Studio