This dataset accompanies the manuscript : M.-R. Goldsmith, CM Grulke, RD Brooks, TR Transue, YM
Isaacs, A Wang, JC Johnson, K Holm, M Reich, J Blackwood, D Vallero, L Phillips, M Phillips, J Wamba

Development of a consumer product ingredient database for chemical exposure screening and pri

Part of FY13 Chemical Safety for Sustainability 2.3.1 products

It includes data on ~1800 chemicals 353 product use categories in nearly 9K products

Technical contact: goldsmith.rocky@epa.gov (919)541-0497
Task Lead: isaacs.kristin@epa.gov

Brief Tabs description:
CPCPDB_RAWDATA: contains all data with specific annotations

CPCPDB_INGREDIENT_FREQUENCY:	***contains a PivotTable or interactive summary - ea*** ***Frequency is the frequency of the chemical mappe*** ***expanded by clicking on the + symbol***
CPCPDB_INGREDIENT_MEAN_PERCENT:	***contains a PivotTable or interactive summary - ea*** ***Mean Percent is the Min/Max average valueof the***

/I Tan, A Frame, R Edwards, PP Egeghy, DT Chang, R Tornero-Velez, K
ugh, R Judson, H Ozkaynak, T Buckley, and CC Dary.

oritization.

:h colored cell can be clicked on to identify a subset of interest
d to a specific product. Each upper level category can be

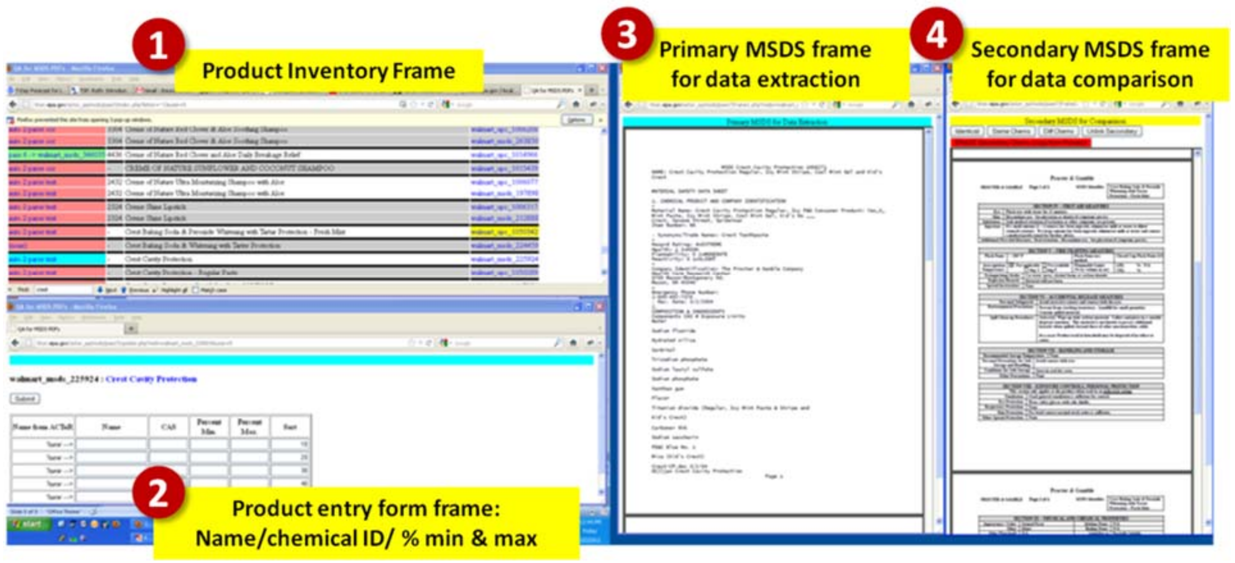
:h colored cell can be clicked on to identify a subset of interest
: chemical mapped to a specific product. Each upper level

Supplementary Information 1 - Detailed database development procedure

A. Identification, retrieval, and extraction: The collection of MSDSs from public repositories was done by scraping the target sites using “wget”, a program that returns the target of a URL. In the case of our source at the time of MSDS retrieval, the URL for each .pdf has a base with simple to guess additions (often called “GET” data). The GET string was quite predictable allowing a PERL script to, when possible, collect as many MSDS files via tools such as “wget” which will return the target of a URL. Many sites have information on the URL (called “GET” data) which can be modified to produce alternate queries resulting in different target files. Some sites offer lists of URLs with one .pdf file per URL, but no schematic way to “guess” the URL pattern for other files, however the list itself may be predictable with, for example, the initial letter of all products as a parameter in the GET string on the URL. In such cases, a PERL script can be written to request each page containing a list of targets, and then the target URLs can be collected to query separately or inside the PERL script. The results are typically hundreds to thousands of .pdf files each with some meta data such as a product name or UPC symbol.

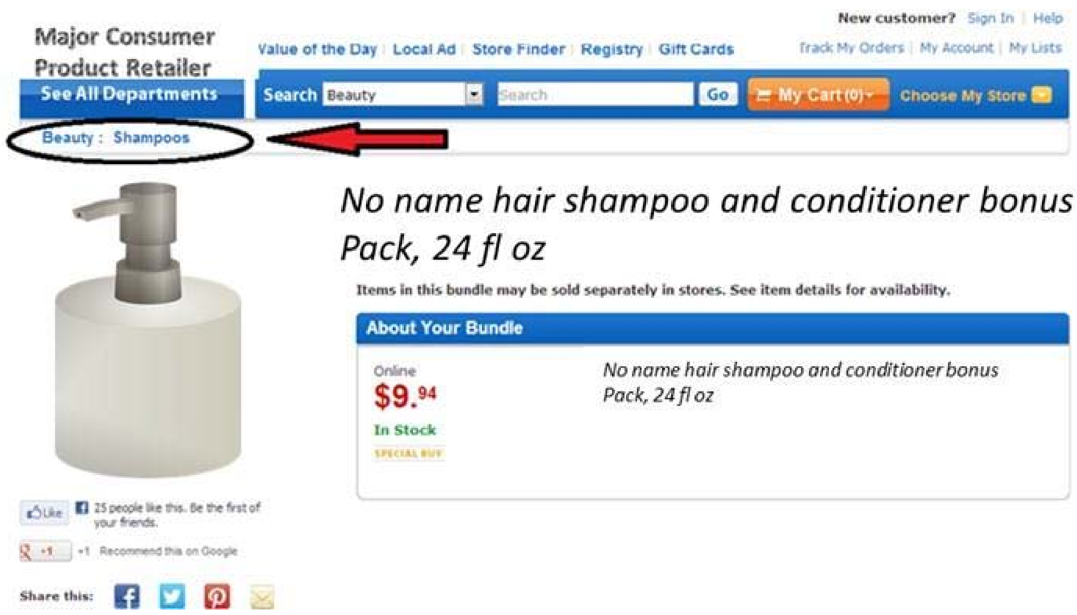
B. Most .pdf files fall into one of two categories: 1) Text which can be parsed (or cut and pasted from a viewer such as Acrobat) and 2) Images resulting from the scanning of printed documents. The latter cannot be directly parsed, but some lend themselves to OCR tools such as Tesseract. The results after modest effort showed generally poor quality (i.e. less than 90% reliable characters), however, this was in many cases adequate to provide a starting point for manual curation. For .pdf files containing defined text, tools such as “pdftotext” (standard on many linux computers) can parse all text quickly and accurately, however, with some frequency the alignment of text on the page is not well represented in the sequential flow of text into a file. The “-layout” option helps to some extent as it attempts to add space characters to align text. A more effective tool might be PDFBox (http://pdfbox.apache.org/) which can detect the coordinates of characters on the page. It requires some effort, however, to be sure that words are kept together as it captures each character separately. The management of data from .pdf files and the MSDS sheets contained therein can be effectively done with a MySQL schema primarily using two tables. One captures information about each product (one per row) assuming that the product has one MSDS file, that file has a URL or other source information, and the product name can be stored as well. A separate table captures chemicals found in products and allows for 0, 1, or more chemicals per product to be recorded. In addition, alternate or successive curation efforts for the same product can have information about each chemical stored along with the date, curation level, and curator information.

C. In order to make the process efficient for the person, a tool was created using a combination of MySQL, PERL (Common Gateway Interface or ,CGI), and HyperText Markup Language, HTML, assuming that the browser could also access the stored .pdf files and display them for inspection. This tool allows the database curator/annotator to view a .pdf file along with meta information on the product, information on the chemicals already found for the product (whether found automatically or from a previous manual curation), and information about any chemicals already found available from a reference data source such as the ACToR database. The curator can then determine which information must be added or updated, make appropriate changes, and save those changes to the MySQL database.


1
Product Inventory Frame
3
Primary MSDS frame for data extraction
4
Secondary MSDS frame for data comparison
2
Product entry form frame: Name/chemical ID/ % min & max

Supporting Information Figure 1: Chemical ingredient and quantitative composition data entry interface built for crowd-sourced curation, optimized for multi-screen end-user, built using a combination of MySQL, PERL (CGI), and HTML.

D. The graphical context for the process of product use category annotation (methods section 2.2) is shown below.



No name hair shampoo and conditioner bonus Pack, 24 fl oz

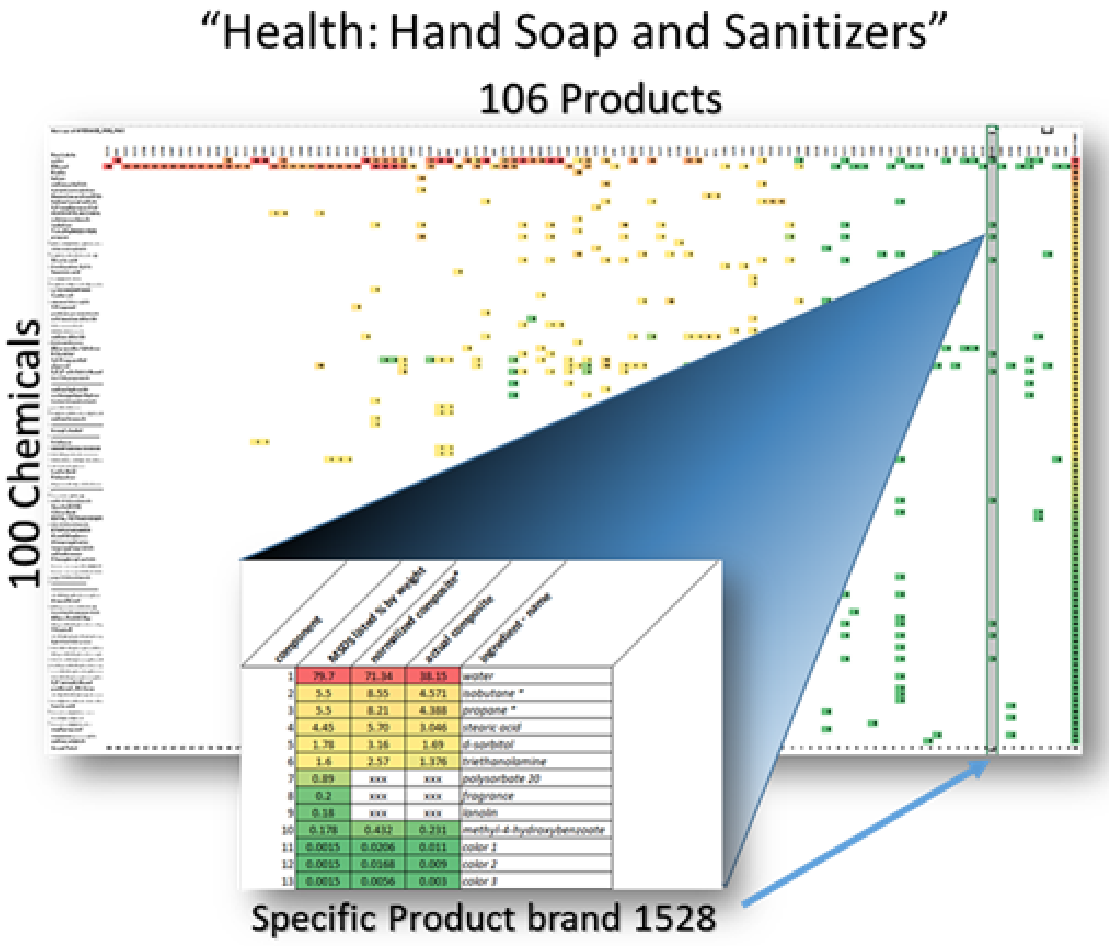
Items in this bundle may be sold separately in stores. See item details for availability.

No name hair shampoo and conditioner bonus Pack, 24 fl oz

Suuporting information Figure 2: Product Use Category information (circled in black with red-arrow) from retailer's e-commerce website, with the breadcrumb containing product use category (top level = Beauty, Sub-category = shampoos) for a fictitious commercial product.

E - Developing and using Generic formulations in the absence of compositional information

The MSDS for product #1528 listed a total of 13 ingredients, (http://goo.gl/74YWX) for which the composite had three ingredients that cannot map to CAS numbers as unique chemical entities. These three ingredients were “fragrance” (potentially 1 of > 3000 different possible fragrances used in commerce), “lanolin” (a mixture of several thousand chemical entities derived/extracted from sheep wool), and “polysorbate 20” (a complex mixture). If we rank order across all identified ingredients by “average % composition” we identify that the rank order of each of the chemicals is “identical” (literally) from highest to lowest % by weight. Furthermore, in the case of propane/isobutane (propellant) a % composition of 11% was listed non-specifically to both combined, our analyses provides a number of 4.6% (isobutane) and 4.4% (propane).



component	MSDS listed % by weight	normalized composite*	actual composite	ingredient - name
1	79.7	71.34	38.15	water
2	5.5	8.55	4.571	isobutane *
3	5.5	8.21	4.388	propane *
4	4.45	5.70	3.046	stearic acid
5	1.78	3.16	1.69	d-sorbitol
6	1.6	2.57	1.376	triethanolamine
7	0.89	xxx	xxx	polysorbate 20
8	0.2	xxx	xxx	fragrance
9	0.18	xxx	xxx	lanolin
10	0.178	0.432	0.231	methyl-4-hydroxybenzoate
11	0.0015	0.0206	0.011	color 1
12	0.0015	0.0168	0.009	color 2
13	0.0015	0.0056	0.003	color 3

Supporting information Figure 3: Example of how a unique product in the “Health:Hand Soap and Sanitizers” category with missing quantitative data (poor data curation with missing % min/max) can still be quantitatively estimated with only the list of ingredients present in the *generic formulation* using the composite data from other “same-in-class” products to generate a relevant ingredient profile. Whether verbatim aggregate estimates (column 3) or normalized estimates (column 2), the rank order of the ingredients derived from composites commensurates with actual ingredient rank order gleaned from the product’s MSDS composition data (column 1).

Supplementary Information 2 - Detailed Use Category Script for Screen-Scraping (session file)

```
<?xml version="1.0" encoding="UTF-8"?>

<scraping-session use-strict-mode="true"><script-instances><script-instances when-to-run="10"
sequence="1" enabled="true"><script><script-text>File inputFile = new File(
&quot;product_names.txt&quot; );

// These two objects are needed to read the file.

FileReader in = new FileReader( inputFile );

BufferedReader buffRead = new BufferedReader( in );

// Read the file in line-by-line. Each line in the text file

// will contain a search term.

while( ( searchterm = buffRead.readLine() )!=null)

{

   // Set a session variable corresponding to the search term.

   session.setVariable( &quot;SEARCHTERM&quot;, searchterm );

   // Get search results for this particular search term.

   session.scrapeFile( &quot;Search&quot;);

}

// Close up the objects to indicate we're done reading the file.

in.close();

buffRead.close();</script-text><name>Walmart--Search Terms</name><language>Interpreted
Java</language></script></script-instances><owner-type>ScrapingSession</owner-
type><owner-name>Walmart</owner-name></script-
instances><name>Walmart</name><notes></notes><cookiePolicy>0</cookiePolicy><maxHTT
PRequests>1</maxHTTPRequests><external_proxy_username></external_proxy_username><
external_proxy_password></external_proxy_password><external_proxy_host></external_proxy
_host><external_proxy_port></external_proxy_port><external_nt_proxy_username></external_
nt_proxy_username><external_nt_proxy_password></external_nt_proxy_password><external_
nt_proxy_domain></external_nt_proxy_domain><external_nt_proxy_host></external_nt_proxy_
host><anonymize>false</anonymize><terminate_proxies_on_completion>false</terminate_pro
xies_on_completion><number_of_required_proxies>5</number_of_required_proxies><originat
or_edition>0</originator_edition><logging_level>1</logging_level><date_exported>April 06,
2012 12:27:29</date_exported><character_set>UTF-8</character_set><scrapeable-files
sequence="-1" will-be-invoked-manually="true" tidy-html="jtidy"><last-scraped-data></last-
scraped-data><URL>http://www.walmart.com/catalog/product.do</URL><last-request></last-
request><name>searchresult</name><extractor-patterns sequence="1" automatically-save-in-
```

session-variable="false" if-saved-in-session-variable="0" filter-duplicates="false" cache-data-set="false" will-be-invoked-manually="false"><pattern-text>&lt;ol id=&quot;crumbs&quot;>

~@DATARECORD@~

&lt;/ol></pattern-text><identifier>crumbs</identifier><extractor-pattern-tokens optional="false" save-in-session-variable="false" compound-key="true" strip-html="false" resolve-relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-data="false" null-session-variable="false" sequence="1"><regular-expression></regular-expression><identifier>DATARECORD</identifier></extractor-pattern-tokens><extractor-patterns sequence="1" automatically-save-in-session-variable="false" if-saved-in-session-variable="0" filter-duplicates="false" cache-data-set="false" will-be-invoked-manually="false"><pattern-text>&lt;li>&lt;a class=&quot;last&quot; href=&quot;~@foo1@~>~@Class1@~&lt;/a>:&lt;/li>

&lt;li>&lt;a class=&quot;last&quot; href=&quot;~@foo2@~&quot;>~@Class2@~&lt;/a></pattern-text><extractor-pattern-tokens optional="false" save-in-session-variable="false" compound-key="true" strip-html="false" resolve-relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-data="false" null-session-variable="false" sequence="3"><regular-expression>[^&lt;>]*</regular-expression><identifier>foo2</identifier></extractor-pattern-tokens><extractor-pattern-tokens optional="false" save-in-session-variable="false" compound-key="true" strip-html="false" resolve-relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-data="false" null-session-variable="false" sequence="1"><regular-expression>[^&lt;>]*</regular-expression><identifier>foo1</identifier></extractor-pattern-tokens><extractor-pattern-tokens optional="false" save-in-session-variable="false" compound-key="true" strip-html="false" resolve-relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-data="false" null-session-variable="false" sequence="2"><regular-expression>[^&lt;>]*</regular-expression><identifier>Class1</identifier></extractor-pattern-tokens><extractor-pattern-tokens optional="false" save-in-session-variable="false" compound-key="true" strip-html="false" resolve-relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-data="false" null-session-variable="false" sequence="4"><regular-expression>[^&quot;]*</regular-expression><identifier>Class2</identifier></extractor-pattern-tokens><script-instances/></extractor-patterns><script-instances><script-instances when-to-run="80" sequence="1" enabled="true"><script><script-text>FileWriter out = null;

try

{

    session.log(&quot;Writing data to a file.&quot;);

    // Open up the file to be appended to.

    out = new FileWriter(&quot;ProductCategories.txt&quot;, true);

out.write(session.getVariable(&quot;SEARCHTERM&quot;)+&quot;\t&quot;);

out.write(dataRecord.get(&quot;Class1&quot;)+&quot;\t&quot;);

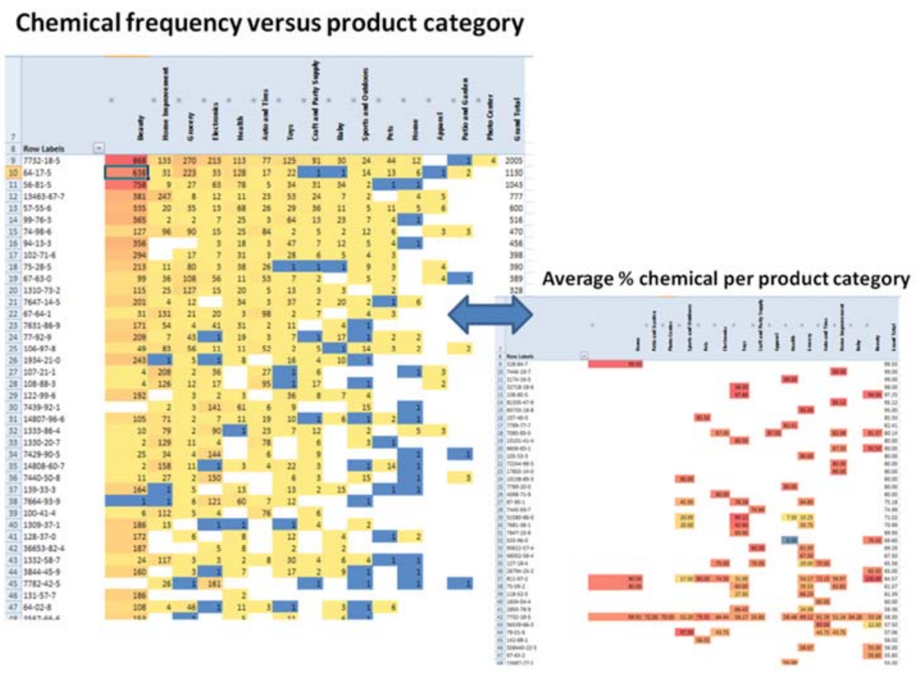
out.write(dataRecord.get(&quot;Class2&quot;)+&quot;\t&quot;);

                    out.write(&quot;\n&quot;);

```
            // Close up the file.
           out.close();
}
catch( Exception e )
{
   session.log( &quot;An error occurred while writing the data to a file: &quot; + e.getMessage()
);
}</script-text><name>Walmart--WriteDataToFile</name><language>Interpreted
Java</language></script></script-instances><owner-type>ExtractorPattern</owner-
type><owner-name>crumbs</owner-name></script-instances></extractor-
patterns><HTTPParameters
sequence="1"><key>product_id</key><type>GET</type><value>~#PRODUCTID#~</value></
HTTPParameters><script-instances><owner-type>ScrapeableFile</owner-type><owner-
name>searchresult</owner-name></script-instances></scrapeable-files><scrapeable-files
sequence="-1" will-be-invoked-manually="true" tidy-html="jtidy"><last-scraped-data></last-
scraped-data><URL>http://www.walmart.com/search/search-ng.do</URL><last-request></last-
request><name>Search</name><extractor-patterns sequence="1" automatically-save-in-
session-variable="false" if-saved-in-session-variable="0" filter-duplicates="false" cache-data-
set="false" will-be-invoked-manually="false"><pattern-text>updateDisplay(&quot;0&quot;,
shelfDiv, gridSpan, listSpan, gridIcon, listIcon, iconPath);
});
&lt;/script>~@somecrap@~&lt;div class=&quot;compare&quot;>&lt;a
href=&quot;/catalog/product.do?product_id=~@PRODUCTID@~&quot; onclick=&quot;j
</pattern-text><identifier>Untitled Extractor Pattern</identifier><extractor-pattern-tokens
optional="false" save-in-session-variable="true" compound-key="true" strip-html="false" resolve-
relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-
data="false" null-session-variable="false" sequence="2"><regular-expression>[\d,]+</regular-
expression><identifier>PRODUCTID</identifier></extractor-pattern-tokens><extractor-pattern-
tokens optional="false" save-in-session-variable="false" compound-key="true" strip-html="false"
resolve-relative-url="false" replace-html-entities="false" trim-white-space="false" exclude-from-
data="false" null-session-variable="false"
sequence="1"><identifier>somecrap</identifier></extractor-pattern-tokens><script-
instances><script-instances when-to-run="80" sequence="1" enabled="true"><script><script-
text>// This will scrape Details page
session.scrapeFile( &quot;searchresult&quot; );</script-text><name>Walmart--
ScrapeSearchResult</name><language>Interpreted Java</language></script></script-
instances><owner-type>ExtractorPattern</owner-type><owner-name>Untitled Extractor
Pattern</owner-name></script-instances></extractor-patterns><HTTPParameters
sequence="4"><key>search_constraint</key><type>GET</type><value>0</value></HTTPPara
meters><HTTPParameters
sequence="3"><key>Find</key><type>GET</type><value>Find</value></HTTPParameters><
```

HTTPParameters
sequence="2"><key>ic</key><type>GET</type><value>16_0</value></HTTPParameters><HT
TPParameters
sequence="1"><key>search_query</key><type>GET</type><value>~#SEARCHTERM#~</valu
e></HTTPParameters><script-instances><owner-type>ScrapeableFile</owner-type><owner-
name>Search</owner-name></script-instances></scrapeable-files></scraping-session>

Supplemental Information 3 – PivotTable of entire CPCPdb (anonymized product names/brands). This will also be included in future releases of ACToR. (http://actor.epa.gov) - About the Consumer Product Chemical Profiles database (CPCPdb) - Product information in the CPCPdb is derived from publicly accessible sources of Material Safety Data Sheets (MSDSs), and future updates will also include information from product labels and information provided directly by manufacturers. The U.S. EPA does not endorse or guarantee the accuracy or completeness of this information. The CPCPdb should not be considered a substitute for obtaining ingredient information either from product labels or directly from the consumer product manufacturers. The Agency makes no expressed or implied warranties, representations or endorsements (including, without limitation, warranties of title or non-infringement, or the implied warranties of fitness of brands for a particular purpose) with regard to any information provided in the CPCPdb The user assumes full responsibility for using these data and all data available through EPA's ACToR (Aggregated Computational Toxicology Resource) site (http://actor.epa.gov) and understands and agrees that the Agency is neither responsible nor liable to anyone for any claim, loss or damage resulting from use of such data.



Chemical frequency versus product category

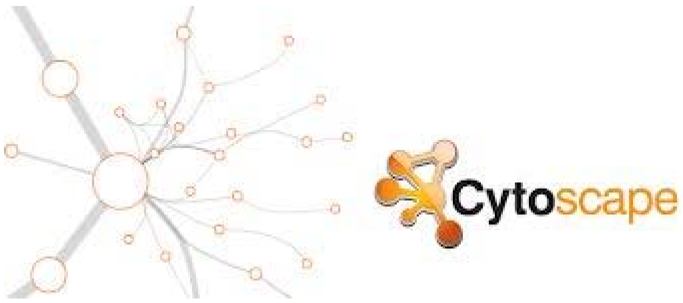
Row Labels	Beauty	Home Improvement	Grocery	Electronics	Health	Auto and Tires	Toys	Craft and Party Supply	Baby	Sports and Outdoors	Pets	Home	Apparel	Patio and Garden	Photo Center	Grand Total
7732-18-5	868	133	270	215	113	77	125	91	30	24	44	12		1	4	2005
64-17-5	638	31	223	33	128	17	22	1	1	14	13	6	1	2		1130
56-81-5	758	9	27	63	78	5	34	31	34	2	1	1				1043
13463-67-7	381	247	8	12	11	23	53	24	7	2		4	5			777
57-55-6	335	20	35	13	68	26	29	36	11	5	11	5	6			600
99-76-3	365	2	2	7	25	3	64	13	23	7	4	1				516
74-98-6	127	96	90	15	25	84	2	5	2	12	6		3	3		470
94-13-3	356			3	18	3	47	7	12	5	4	1				456
102-71-6	294		17	7	31	3	28	6	5	4	3					398
75-28-5	213	11	80	3	38	26	1	1	1	9	3		4			390
67-63-0	99	36	108	56	11	53	7	2		5	7		4	1		389
1310-73-2	115	25	127	15	20	5	13	3	3		2					328
7647-14-5	201	4	12		34	3	37	2	20	2	1	6				
67-64-1	31	131	21	20	3	98	2	7		4	3					
7631-86-9	171	54	4	41	31	2	11		4	1						
77-92-9	209	7	43	1	19	3	7	1	17	1	2	2				
106-97-8	49	83	56	11	11	52	2	5	1	14	3	2		2		
1934-21-0	243	1	5	1	8		16	4	10	1						
107-21-1	4	208	2	36		27	1	6				1	3			
108-88-3	4	126	12	17		95	1	17		1			2			
122-99-6	192		3	2	3		36	8	7	4						
7439-92-1		2	3	141	61	6	9			15		1				
14807-96-6	105	71	2	7	11	19	10	1	6	1	2	1				
1333-86-4	10	79	2	90	1	23	7	12		2		5	3			
1330-20-7	2	129	11	4		78		6		3	1					
7429-90-5	25	34	4	144		6		9				1		1		
14808-60-7	2	158	11	1	3	4	22	3		1	14	1				
7440-50-8	11	27	2	150			6	3		15		1		3		
139-33-3	164	1	5		13		13	2	15		1	1				
7664-93-9	1	1	6	121	60	7	12			1						
100-41-4	6	112	5	4		76		6								
1309-37-1	186	13		1	1		1	4		2						
128-37-0	172		6		8		12		4		1	2				
36653-82-4	187			5	8		2		2							
1332-58-7	24	117	3	3	2	8	30	4	6	4	1	1				
3844-45-9	160		3	1	7		17	2	9	1		1				
7782-42-5		26	1	161				1		1		1		1		
131-57-7	186				2											
64-02-8	108	4	46	1	11	3	1		3	1	6					

Average % chemical per product category

Table 4: Screenshot of chemical CAS numbers (rows) and top level consumer product use categories (columns) frequency in products (left) and % composition (right) available in the spreadsheet in supporting information.

CPCPDB_DATA_SUPPORTINGINFORMATION.xlsx

Supplementary Information 4 – CytoScape network file with chemicals linked to product use categories.



SEE: CPCPDB_NETWORK_DIAGRAM_SUPPORTINGINFORMATION_4.CYS downloadable from http://goo.gl/hwvPyg