WHY AN OBJECT-CENTRIC VIEW WILL BE KEY FOR FUTURE AI SYSTEMS

WHITE PAPER

FROM RELATIONAL TABLES TO “OBJECT ANALYTICS”

Xplain Data GmbH
Grünlandstr. 27
85604 Zorneding
Germany
info@xplain-data.com
xplain-data.com

WHAT TO EXPECT

SUMMARY

Relational databases are a key element in the today's world of data. They are a perfect solution for what they have been built for: transactionally managing data. For that they split an object into different entities and store different entities in different tables.

Once, however, an object is split into entities and distributed across different tables, it is hard to analyze the object "as a whole". The object in focus of analysis might be "The Patient" with different data streams attached to it (diagnoses, treatments, prescriptions etc.).

If a patient's diagnoses and treatments are scattered across different tables, it will be difficult to analyze them in relation to each other. An example: you want to know how patients are treated after a COVID-19 diagnosis. It requires to collect all diagnoses and treatments for each patient (millions of them!) from different tables. Obviously, having all information for each patient already stored "jointly" would help. When repeatedly scanning millions of patients to compute statistical data, we don't need to collect all patient instances repeatedly from different tables. I.e., we need the object "Patient" accessible as a whole.

That's what Object Analytics facilitates. For the same reason that relational databases are the best solution for transactional data management, they are suboptimal for holistic analytics. Vice versa, our Object Analytics database is not meant for transactionally managing data, but it is the best solution for "holistic analytics". Future analyses will require such a holistic view of your company's focus object - and thus an object-centric representation of data will become mandatory.

This White Paper explains our "Object Analytics" paradigm and typical analytic operations on whole objects that are supported. The Xplain Object Explorer (XOE) builds on this and implements a novel interactive frontend that allows complex objects to be explored analytically. Advanced analytics and future Artificial Intelligence algorithms require such a holistic view. An example is our Causal Discovery approach. You will learn why understanding cause and effect based on observational data will be a key element of future AI systems, and why holistic information is essential for this.

OBJECT ANALYTICS VS. RELATIONAL TABLES

WHAT RELATIONAL DATABASES ARE MEANT FOR – AND WHAT NOT

First step in designing a relational data model is defining the entities of the model, and how they relate to each other (the keys). Different entities are then stored in different tables. This, in particular, facilitates transactional operations, i.e., append, update, delete rows in a table. Adding a row in a table is just a millisecond – as only this single table is touched. Also, from an analytics point of view, such a relational model is very generic – no assumptions need to be made about what the object in focus of the analysis is.

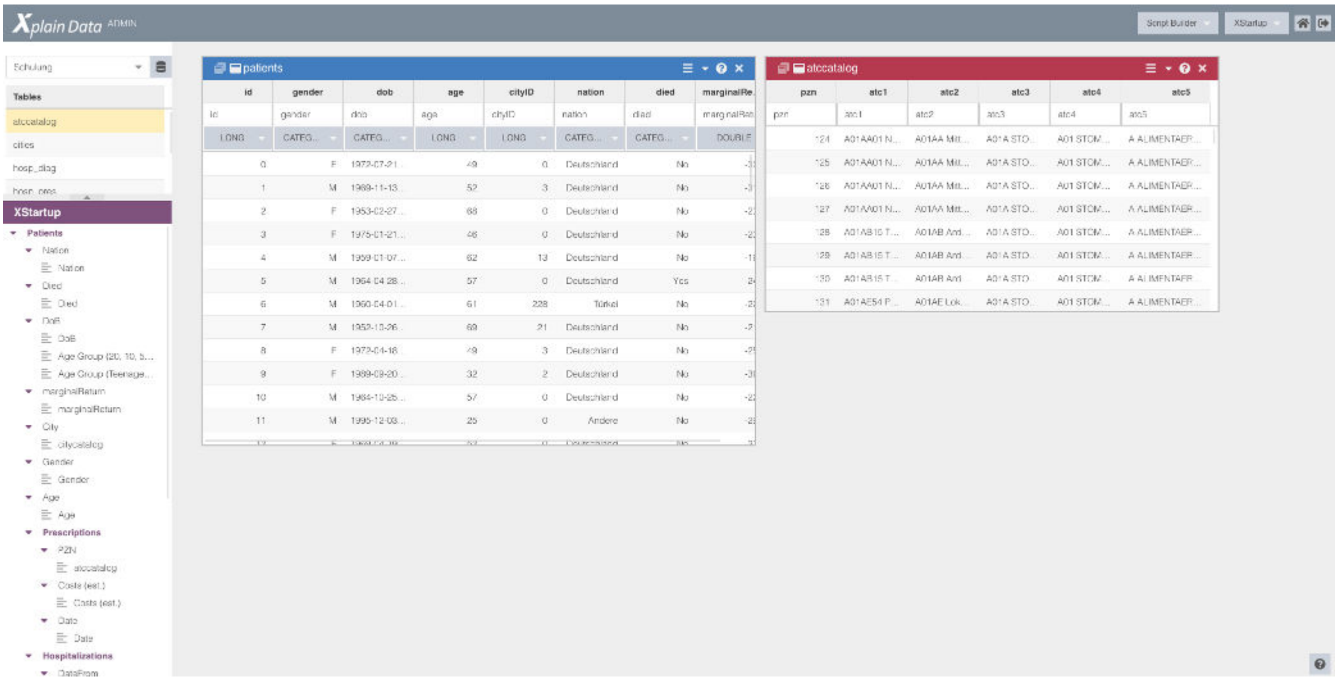
But: in 99 % of all analytical scenarios, a specific object is the focus of attention. In healthcare it is typically “The Patient”, in industry it’s “The Machine” or “The Part” that is manufactured, in CRM it’s “The Customer”.

Whenever a particular object is the focus of your analysis, it is a good idea to organize the data in a way that all data can be collected very quickly for each object instance.

HOW OBJECT ANALYTICS DIFFERS FROM RELATIONAL DATABASES

This is exactly what our Object Analytics Database does:

1. Pull data from a relational database or other data sources (see figure below)
2. Re-organize it in an “object-centric” manner
3. Scan millions of object instances in a second - to execute complex analytical operations which required the object as whole.



patients

id	gender	dob	age	cityID	nation	died	marginalRe...
id	gender	dob	age	cityID	nation	died	marginalRet
LONG	CATEG...	CATEG...	LONG	LONG	CATEG...	CATEG...	DOUBLE
0	F	1972-07-21...	49	0	Deutschland	No	-3
1	M	1969-11-13...	52	3	Deutschland	No	-3
2	F	1953-02-27...	68	0	Deutschland	No	-2
3	F	1975-01-21...	46	0	Deutschland	No	-2
4	M	1959-01-07...	62	13	Deutschland	No	-1
5	M	1964-04-28...	57	0	Deutschland	Yes	2
6	M	1960-04-01...	61	228	Türkei	No	-2
7	M	1952-10-26...	69	21	Deutschland	No	-2
8	F	1972-04-18...	49	3	Deutschland	No	-2
9	F	1989-09-20...	32	2	Deutschland	No	-3
10	M	1964-10-25...	57	0	Deutschland	No	-2
11	M	1995-12-03...	25	0	Andere	No	-2

atccatalog

pzn	atc1	atc2	atc3	atc4	atc5
pzn	atc1	atc2	atc3	atc4	atc5
124	A01AA01 N...	A01AA Mitt...	A01A STO...	A01 STOM...	A ALIMENTAER...
125	A01AA01 N...	A01AA Mitt...	A01A STO...	A01 STOM...	A ALIMENTAER...
126	A01AA01 N...	A01AA Mitt...	A01A STO...	A01 STOM...	A ALIMENTAER...
127	A01AA01 N...	A01AA Mitt...	A01A STO...	A01 STOM...	A ALIMENTAER...
128	A01AB15 T...	A01AB And...	A01A STO...	A01 STOM...	A ALIMENTAER...
129	A01AB15 T...	A01AB And...	A01A STO...	A01 STOM...	A ALIMENTAER...
130	A01AB15 T...	A01AB And...	A01A STO...	A01 STOM...	A ALIMENTAER...
131	A01AE54 P...	A01AE Lok...	A01A STO...	A01 STOM...	A ALIMENTAER...

Figure 1 - With the Xplain Data Admin Tool (XAT), you can quickly and easily import data from your data sources.
You can then attach this data into your object tree.

An example: let’s assume a health insurance company with root-object “The Patient” and 10 million enrolled patients. This root object has several sub-objects, such as diagnoses, prescriptions, treatments, hospital cases. Many of these sub-objects have recursive sub-objects (sub-sub-objects).

A hospital case in itself is an object with further sub-objects. Within a hospital case you usually find procedures, measurements of clinical parameters, cost items … When adding all data to the root object, you typically end up in an object model with more than 20, sometimes even 50 sub-objects.

Typically, a sub-object corresponds to a stream of events. A patient’s diagnoses, for example, are a series of events, each with a timestamp. In our health insurance example, let´s assume that there are longitudinal data for each patient for up to 5 years (all their diagnoses, prescriptions, treatments, procedures …). With 10 million patients, you quickly get to a few billion events stored in sub-objects of the root object.

Based on this data, you may analyze out how patients are treated after a COVID-19 diagnosis and what other diagnoses follow. In a relational database, you would have to develop some “ugly” SQL code. Later you will see that this is quite easy to do in our Object Analytics environment - just a few lines of code that are easy to understand.

The experienced SQL developer might object and still prefer SQL. But even if you have such a competent SQL-expert in your department, long runtimes of such complex SQL are typically the result – far from supporting an interactive analysis process.

Remember: for the above task of analyzing diagnoses related to treatments, we need to “join” two different tables via the patient key (a foreign key in each of them), where each table may contain billions of events.

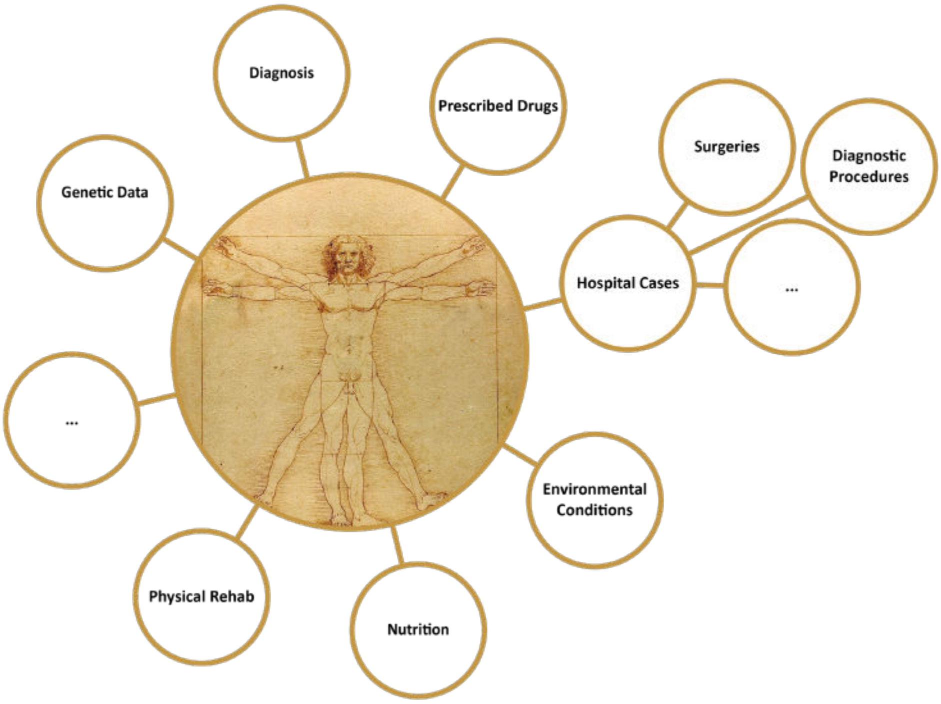


Figure 2 - Example of a complex, real-world object: “The Patient” with various data streams attached to it. “Object Analytics” means being able to analyze the different data areas in relation to each other.

ITERATING OBJECTS AND "OBJECT MAP REDUCE"

An SQL statement which aggregates statistics will cycle through all rows in a table (iterate all rows) to collect the requested statistics. In Object Analytics, we do not iterate rows in a table, we iterate all instances of an object (all patients). The Object Analytics Database is designed to make this pass through all objects extremely fast. Each object already contains all patient-related information, e.g., all diagnoses and treatments. There is no need to expensively collect all records related to a patient from different tables. Once you have an object at hand as a whole, it is easy to implement complex operations on this object to compute the required statistics.

And this can be done very efficiently in parallel with a number of threads on a multi-core machine. As each object instance (each patient) can be handled independently of the others, the load can be easily distributed across different cores: The first million patients go on core 1, the next million on core 2, … Even for very big machines with 256 cores, all cores can be fully utilized. Once the results on sub-sets of patients are computed, they are consolidated into a joint result (the final reduce step).

For implementing a custom algorithm, the developer needs to specify only two methods:

1. how a single object instance contributes to the required statistics
2. how results on sub-sets are consolidated to a joint result.

Technically, the developer needs to implement those two methods required by our Object Map Reduce interface. Once those two methods are defined, the Object Analytics engine will use those definitions to execute the specified algorithm in a fully parallel manner.

EXAMPLES FOR OBJECT ANALYTICS

EXAMPLES FOR OBJECT-CENTRIC OPERATIONS

The "relative time axis" is an example of an artifact defined in an object model. It computes the time difference between events in two different branches of the object tree: for example, the relative time of treatments in relation to the first COVID-19 diagnosis. In our example this relative time allows to select the first weeks after COVID-19 and statistically see which drugs are most frequently used in this relative time period.

The relative time axis is an example of an artifact which operates across different branches of an object tree (across different sub-objects, e.g., across diagnoses and prescriptions).

Aggregations along the edges of the object tree is another example. A simple one is a "COUNT" for each patient, e.g., the count of anti-depressive drugs prescribed in a defined time period. This count aggregates data from the object-level prescriptions (sub-object) to the level of patients (root object level). Similar aggregations may be defined along any branches of an object tree and with arbitrary aggregation functions. A more complex example might be: sum total anti-depressant dose for each patient 6 months after a COVID-19 diagnosis.

There are several such “object-centric artifacts” that relate different sub-objects to each other:

- a “rank definition” – it assigns a consecutive number to each event within a class of events

- “sequence aggregations” – they allow to get the answer to questions like “what is the first, second, third event of a specific type after a COVID-19 diagnosis”

Xplain Data is continuously expanding these Object Analytics capabilities.[1]

DEFINING OBJECT ANALYTICS ARTIFACTS: PROGRAMMING INTERFACES, E.G. PYTHON

Different interfaces are available to define Object Analytics artifacts and perform queries on available dimensions in the object tree:

- **JavaScript:** this makes it easy to create web-based analytical applications. Our Xplain Object Explorer (see section below) is such an example of a web-based application. The JavaScript interface targets the web-developer.

- **Python:** to do all those Object Analytics operations from within our Python environment. Let’s see a few lines of Python code required for the above example: treatments after COVID-19.

```
xsession.run({
        “method”: “addRelativeTimeDimensions”,
“name”: “Time relative to Covid-19 diagnosis”,
        “timeDimension”: {
                “dimension”: “Date of Therapy”,
                “object”: “Therapy”
        },
“referenceTimeDimension”: {
                “dimension”: “Date of diagnosis”,
                “object”: “Diagnoses”
        },
“referenceEventSelections”: [{
                “attribute”: {
                        “attribute”: “ICD”,
                        “dimension”: “ICD”,
                        “object”: “Diagnoses”
                },
                “selectedStates”: [“COVID-19”]
        }]
})
```

Figure 3 - Example call for creating a relative time axis: It creates a new dimension with name „Time relative to COVID-19 diagnosis“ for all events in the “Therapy” object. The required time and reference time dimension are given by the corresponding arguments. The type of events which define the reference point is given by the argument “referenceEventSelections”.

[1] Our Object Analytics approach is protected by our US patent 11,194,811 B2 and EU patent EP3460680.

This call only declares the relative time axis – no computation is done at that point of time. The heavy lifting comes only as soon as a query is executed which refers to the relative time axis. In the below example you see such a query.

How much more SQL code would you need to do the same? And if you get it done, will it execute quickly on a multi-core machine? On a machine with 256 cores and for the above 10 million patients with a couple of billion events (diagnoses and treatments) this query will execute in around a second. During this second all cores will be processing at full load – i.e., all resources will be optimally exploited.

```
result_dataframe = xsession.execute_query({
        “aggregations”: [{
                   “object”: “Therapy”,
                   “type”: “COUNT”
                }],
        “groupBys”: [{
                “attribute”: {
                        “object”: “Therapy”,
                        “dimension”: “Treatment type”,
                        “name”: “Treatment type”
                        }
                }],
        “selections”: [{
                “attribute”: {
                        “object”: “Diagnoses”,
“dimension”: “Relative Time”,
“name”: “Relative Time”
                        },
                “selectedStates”: [“[0w,4w[“]
                }]
})
```

Figure 4 - Python example of a query using the above declared relative time: This query selects all treatments in the first 4 weeks after a previous COVID-19 diagnosis and counts the number of treatments grouped by type of treatment. Results are immediately converted into a Pandas data frame, a well-known data structure in Python. As such, it may immediately be used for further processing or charting.

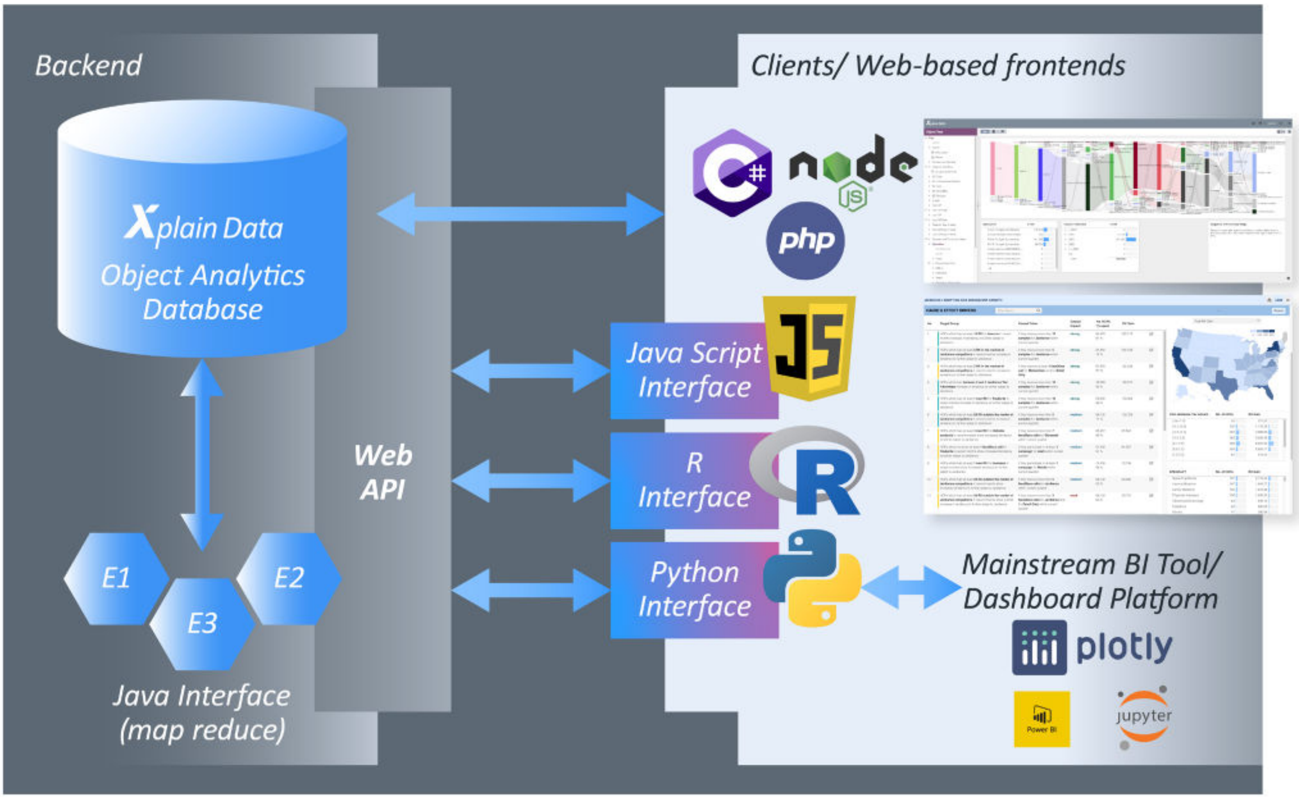
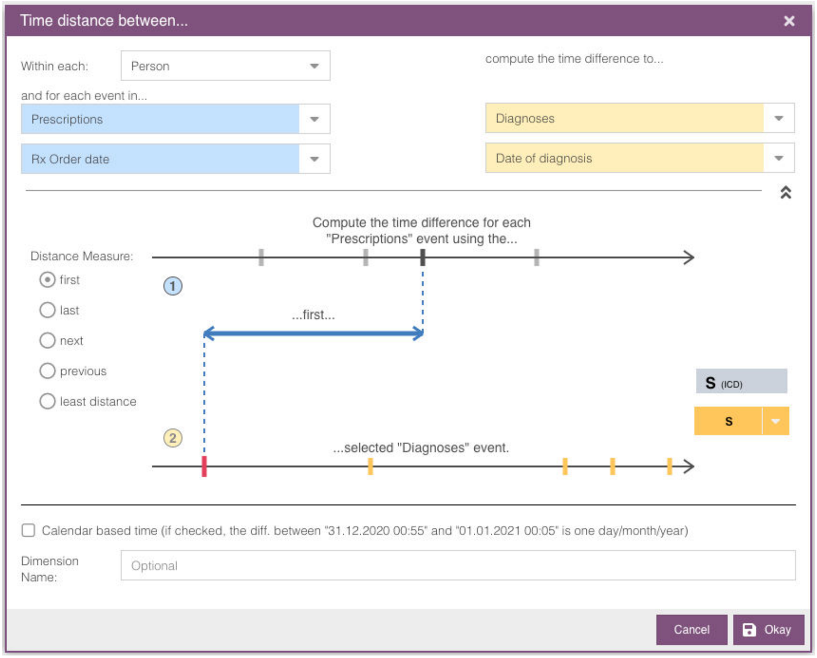


Figure 5 - General architecture: The Object Analytics database as the backend, interfaces and applications using these interfaces.

THE XPLAIN OBJECT EXPLORER

The Xplain Object Explorer (XOE) builds upon the above interfaces. It implements an interactive usability concept to statistically explore a complex object in an agile way. You can set any selections; those selections “propagate” along the edges of the object tree, and you will see how they affect other parts of the object tree. The XOE is a web-based application - all you need is an up-to-date browser. Just enter the correct URL to connect to an Object Analytics database.

Within the XOE you may define any of the above-mentioned artifacts, for example a relative time axis. Figure 6 below shows the corresponding menu which allows you to set the parameters of the relative time axis.



Time distance between...

Within each: Person

and for each event in...

Prescriptions

Rx Order date

compute the time difference to...

Diagnoses

Date of diagnosis

Compute the time difference for each "Prescriptions" event using the...

Distance Measure:

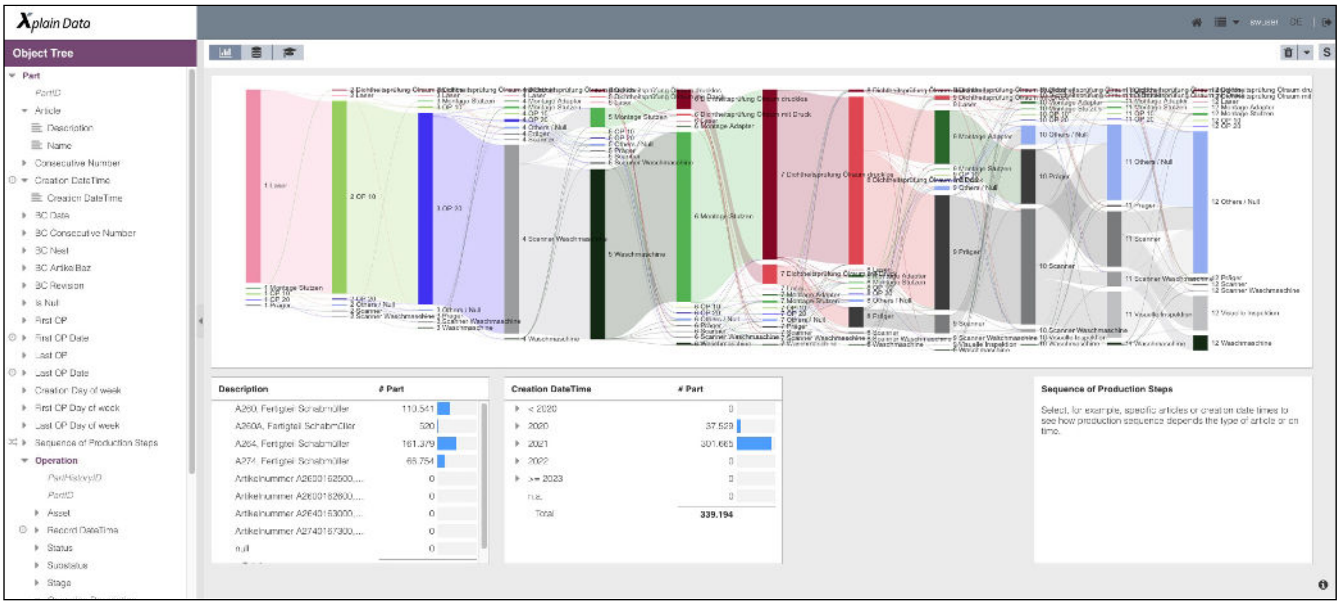
- ● first
- ○ last
- ○ next
- ○ previous
- ○ least distance

□ Calendar based time (if checked, the diff. between "31.12.2020 00:55" and "01.01.2021 00:05" is one day/month/year)

Dimension Name: Optional

Figure 6 - Definition of a relative time axis: For all prescriptions, this time axis computes the time difference to the specified type of diagnoses (to the first one if the patient has multiple of those diagnoses).

Further UI-elements permit the creation of other sorts of Object Analytics artifacts, e.g., aggregation dimensions or rank or sequence definitions. Figure 7 below shows a further example – in this case from the manufacturing industry. This analysis allows to explore the sequence of production steps, e.g., prior to failure events.



Description	# Part
A260, Fertigteil Schabmüller	110.541
A260A, Fertigteil Schabmüller	520
A264, Fertigteil Schabmüller	161.379
A274, Fertigteil Schabmüller	66.754
Artikelnummer A2600162500,...	0
Artikelnummer A2600162600,...	0
Artikelnummer A2640153000,...	0
Artikelnummer A2740167300,...	0
null	0

Creation DateTime	# Part
< 2020	0
2020	37.529
2021	301.665
2022	0
>= 2023	0
n.a.	0
Total	339.194

Sequence of Production Steps

Select, for example, specific articles or creation date times to see how production sequence depends the type of article or on time.

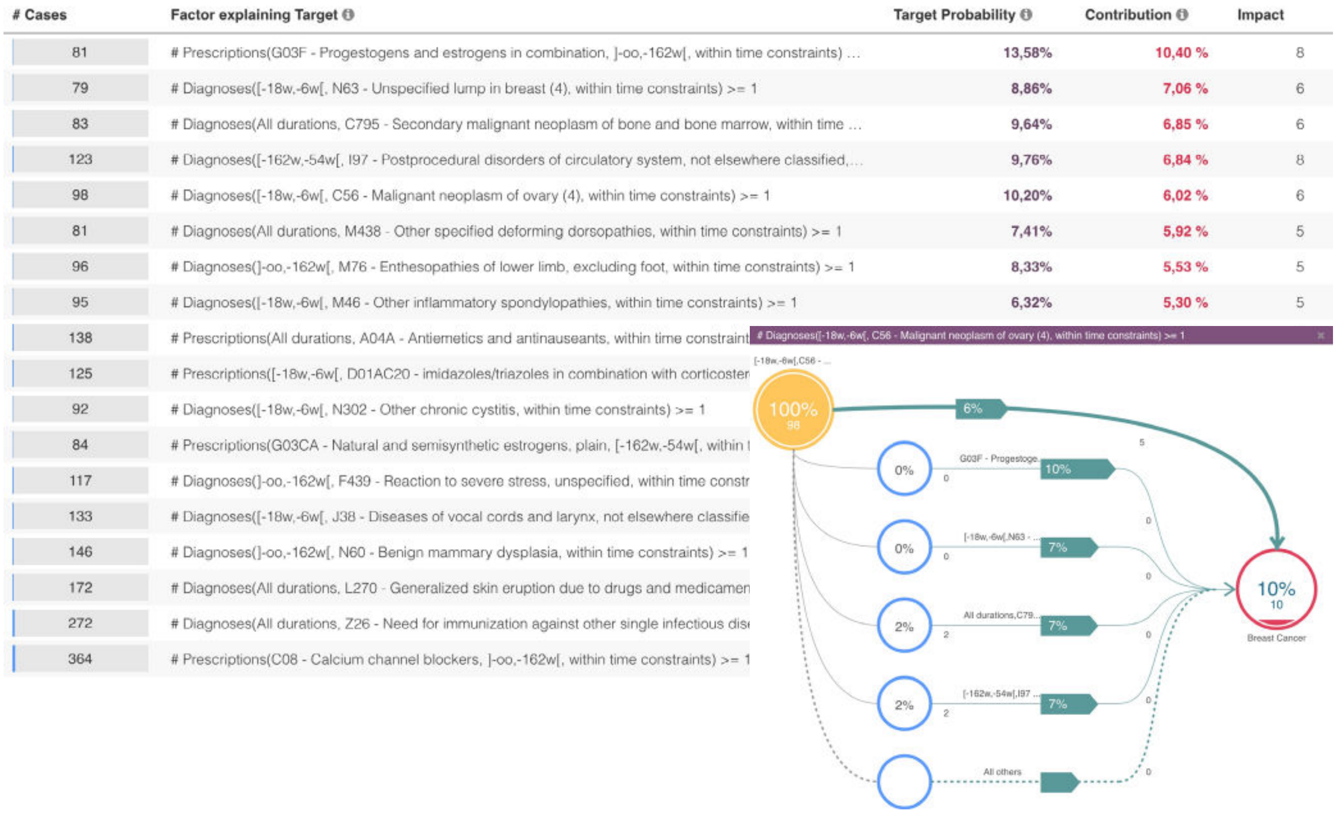
Figure 7 - Analysis of the sequence of production steps. Using the relative time axis, you may as well explore the production process relative to a failure event. The analysis is fully interactive, e.g., select different types for produced parts or different month to see how the process flow differs with that.

[2] This example was kindly provided by Schwäbische Werkzeugmaschinen GmbH (SW).

All artifacts used in the above analysis can also be defined programmatically, e.g., by means of the Python or JavaScript interface. Do the entire analysis in Python or use the XOE in combination with Python. The interplay between the XOE as a graphical user interface and working on a programmatical level is highly supported. If, for example, you have defined an artifact in the XOE (e.g., a relative time axis), simply look up its definition in XOE, and, in terms of a few lines of Python code, copy this definition and paste it into your Python code.

CAUSAL DISCOVERY - AND WHY FUTURE ALGORITHMS REQUIRE OBJECTS AS A WHOLE

Object Analytics aims at analyzing complex objects across all data attached to it. It facilitates a holistic view to a complex object. Understanding cause and effect based on observational data requires such a holistic view. The correlation between two variables A and B can easily be computed – it requires these two data fields only. As we all know, however, correlation does not equal causation. For example, the two variables “gray hair” and “wearing glasses” are correlated: amongst persons with glasses an increased fraction also has gray hair.



# Cases	Factor explaining Target	Target Probability	Contribution	Impact
81	# Prescriptions(G03F - Progestogens and estrogens in combination,]-oo,-162w[, within time constraints) …	13,58%	10,40 %	8
79	# Diagnoses([-18w,-6w[, N63 - Unspecified lump in breast (4), within time constraints) >= 1	8,86%	7,06 %	6
83	# Diagnoses(All durations, C795 - Secondary malignant neoplasm of bone and bone marrow, within time …	9,64%	6,85 %	6
123	# Diagnoses([-162w,-54w[, I97 - Postprocedural disorders of circulatory system, not elsewhere classified,…	9,76%	6,84 %	8
98	# Diagnoses([-18w,-6w[, C56 - Malignant neoplasm of ovary (4), within time constraints) >= 1	10,20%	6,02 %	6
81	# Diagnoses(All durations, M438 - Other specified deforming dorsopathies, within time constraints) >= 1	7,41%	5,92 %	5
96	# Diagnoses(]-oo,-162w[, M76 - Enthesopathies of lower limb, excluding foot, within time constraints) >= 1	8,33%	5,53 %	5
95	# Diagnoses([-18w,-6w[, M46 - Other inflammatory spondylopathies, within time constraints) >= 1	6,32%	5,30 %	5
138	# Prescriptions(All durations, A04A - Antiemetics and antinauseants, within time constraint			
125	# Prescriptions([-18w,-6w[, D01AC20 - imidazoles/triazoles in combination with corticoster			
92	# Diagnoses([-18w,-6w[, N302 - Other chronic cystitis, within time constraints) >= 1			
84	# Prescriptions(G03CA - Natural and semisynthetic estrogens, plain, [-162w,-54w[, within t			
117	# Diagnoses(]-oo,-162w[, F439 - Reaction to severe stress, unspecified, within time constr			
133	# Diagnoses([-18w,-6w[, J38 - Diseases of vocal cords and larynx, not elsewhere classifie			
146	# Diagnoses(]-oo,-162w[, N60 - Benign mammary dysplasia, within time constraints) >= 1			
172	# Diagnoses(All durations, L270 - Generalized skin eruption due to drugs and medicamen			
272	# Diagnoses(All durations, Z26 - Need for immunization against other single infectious dis			
364	# Prescriptions(C08 - Calcium channel blockers,]-oo,-162w[, within time constraints) >= 1			

Diagnoses([-18w,-6w[, C56 - Malignant neoplasm of ovary (4), within time constraints) >= 1

Figure 8 - Example results from a Causal Discovery analysis: Factors which seem to drive breast cancer.

But glasses do not cause your hair to become gray. The reason for the observed correlation is a common cause - age: the older you get, the more likely you need to wear glasses, and many of us also get gray hair.

Many of those common causes might be existing – all potential hypotheses need to be evaluated. In a sense, causation is the opposite to correlation. Correlation requires only the two corresponding data fields, causation requires “anything else” about the object in focus of analysis. If no common causes (confounders) can be found to explain an observed correlation between two variables, only then we might suspect a direct cause-and-effect relationship.

In other words, understanding potential cause and effect relationships requires a holistic view to the object in focus of analysis.

And – by what has been detailed in this White Paper – that exactly is what our Object Analytics approach facilitates. For more details on Causal Discovery see our other White Papers (www.xplain-data.com).

The analysis of complex real-world objects and future Artificial Intelligence algorithms will need an object-centric representation of data. Object Analytics constitutes a tremendous opportunity to develop novel algorithms for the next generation of Artificial Intelligence.

Xplain Data
Discover Causality

Xplain Data GmbH
Grünlandstr. 27
85604 Zorneding
Germany
info@xplain-data.com
xplain-data.com