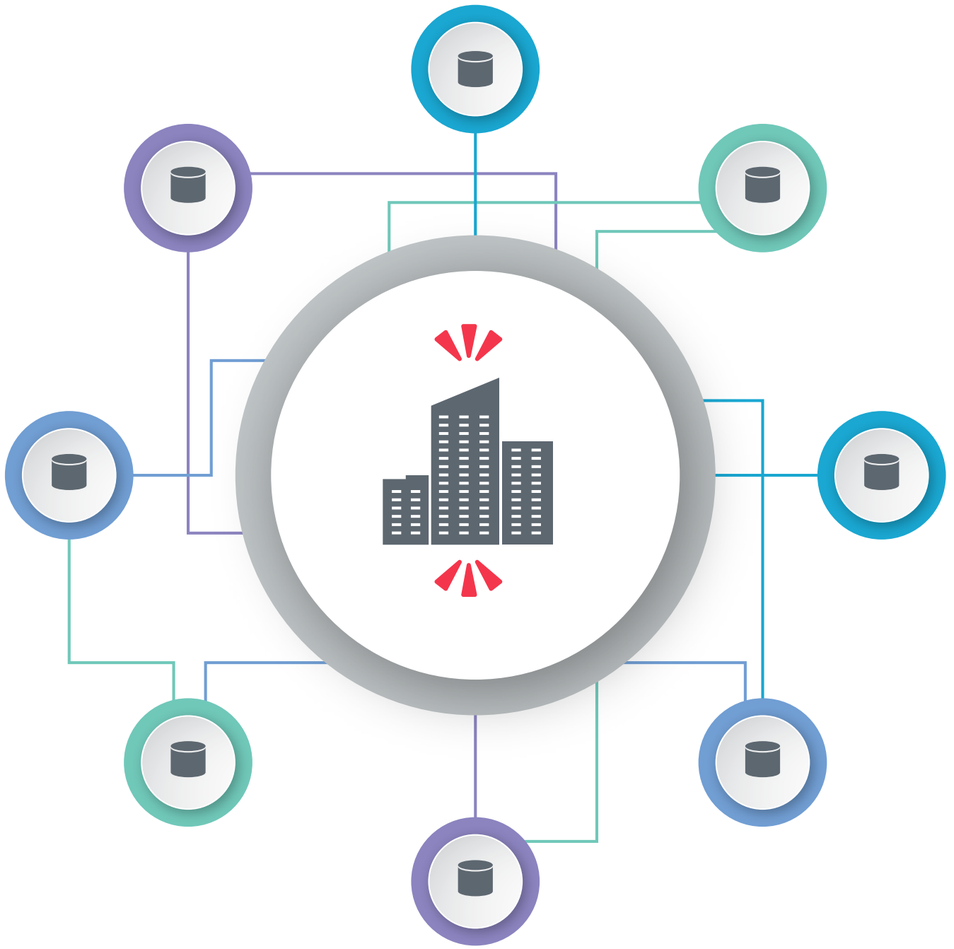
MarkLogic®

Escape the Matrix

Free your data using a multi-model database

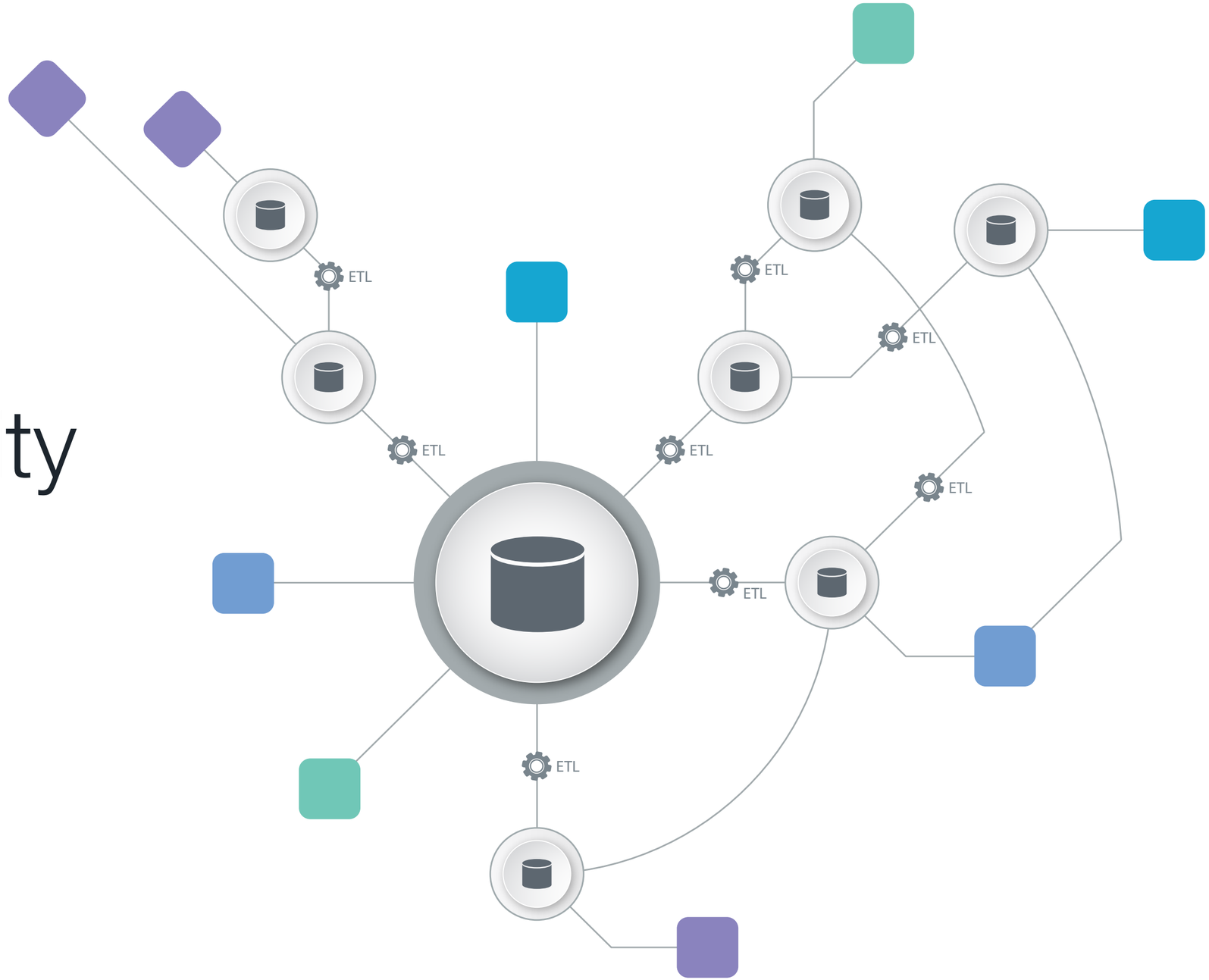
You need a 360 view of your business

You have more customer data coming from more sources than ever before. But chances are, it is spread across silos and stored in relational data models, including registration databases, fulfillment and CRM systems, and ad serving platforms. Unless you can quickly integrate this data to create a 360 view of your customer, your customer data is not actionable.

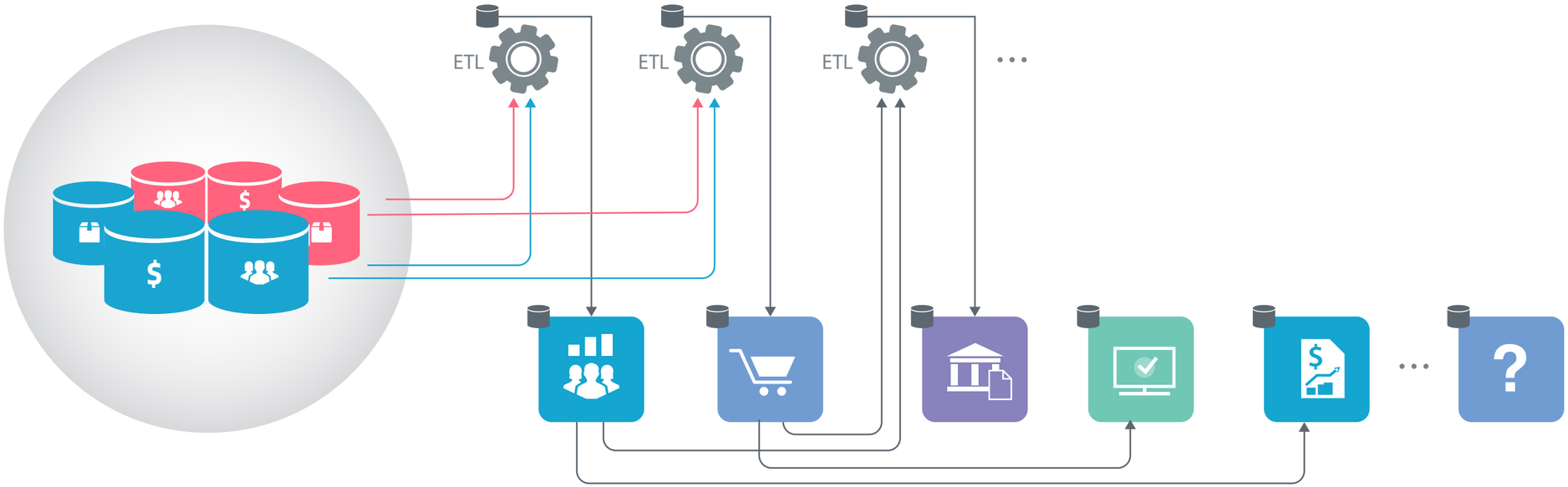


ETL kills agility

Every time you merge data from different relational databases, you must extract, transform and load (ETL) the data, which is a slow, complex and expensive process.



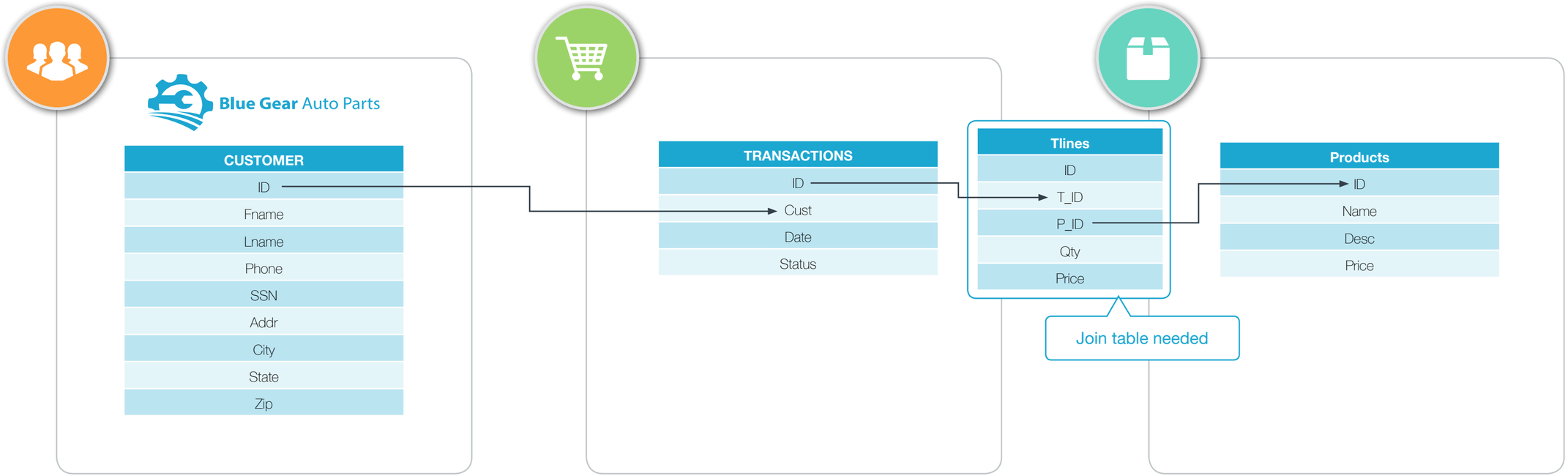
Data is constantly changing



Every new application requires a unique view of your data. A merger or acquisition changes the amount and type of data you manage. As compliance rules change, so too must your approach to governing data. And as digital transformation remakes your business, data must be available in real time. Relational data models are not designed for business agility.

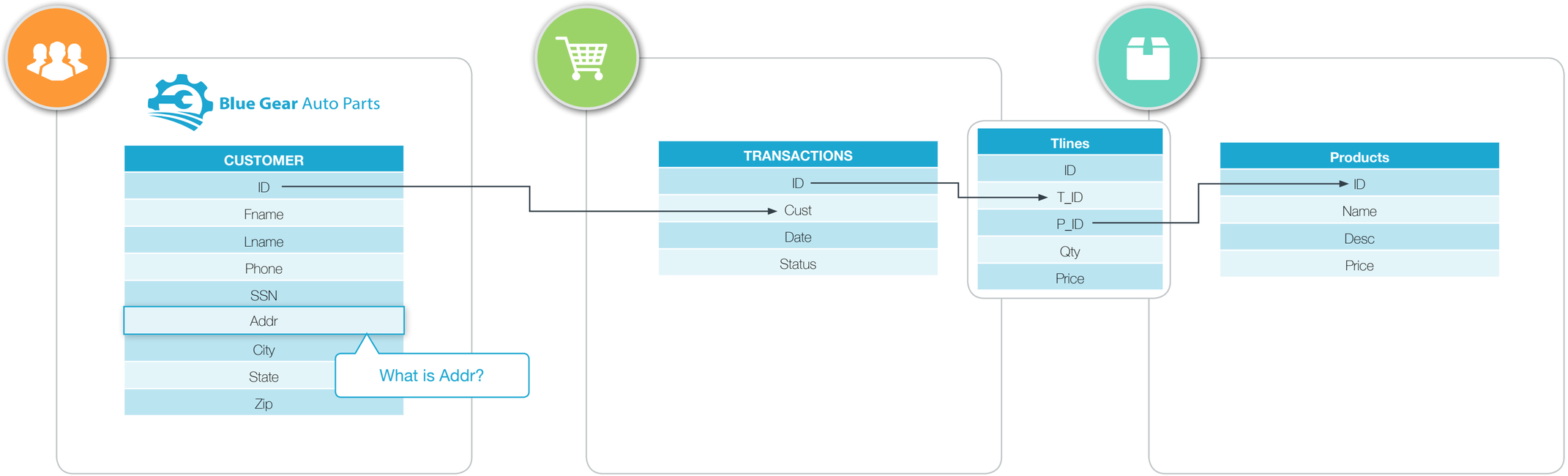
In relational models, even the simple gets complex

Consider the simple entity diagram above: A customer completing a transaction involving a product. Conceptually, this is a very simple entity relationship. But due to the rigidity of relational databases, even simple entity relationships become complex. And in a relational database, similar information can be modeled differently, which makes integrating data difficult.



Not all joins are meaningful

Looking at Blue Gear Auto Part's simple schema, you will notice a problem. It features a join table, "Tlines," that does not map to our conceptual understanding of the transaction. Since transactions can include multiple products—and products can be included in multiple transactions—separate join tables are necessary to store product information. Join tables exist because relational databases force data into columns and rows.



Schemas are not always intuitive

Relational schemas are also difficult to intuitively understand. For example, what does “Addr” represent in the customer table above? You would need to examine this table closely to understand what “Addr” means. As information changes, the rigid nature of relational data models presents more problems. For example, let’s say you need to add a second address or phone number to a customer’s record. In a relational model, this requires new join tables, which further complicates your schema.

Rigid schemas often lead to bad data

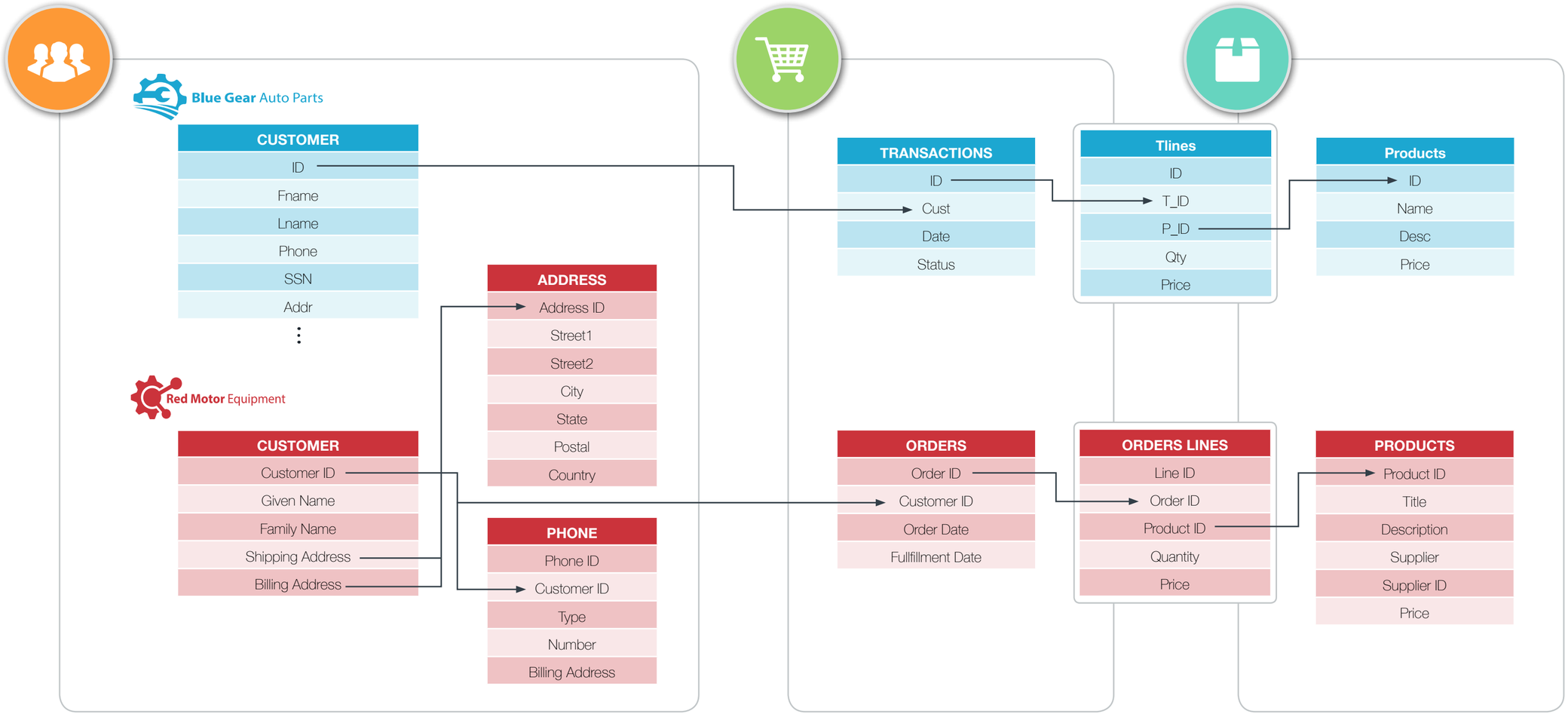
This schema allows for only one phone number because Blue Gear Auto Parts did not account for mobile phones. Instead of redesigning the schema, developers forced two phone numbers into a field designed for one phone number.

ID	Fname	Lname	Phone	SSN	Addr	City	State	Zip
1001	Paul	Jackson	**415-555-1212 | 415-555-1234**	123-45-6789	123 Avenue Road	San Francisco	CA	94111

The inflexible nature of relational data models creates another dilemma: Once you design a schema, the only way to change it is to break it apart and start over. So how do you create a schema that accommodates your future needs? What if your business changes and you need to model your data differently? You can choose the slow and expensive route and design a new schema and ETL your data. Or you can shortcut that process and force your data into your existing schema. But that would cause a data quality problem, because your schema would not match your data.

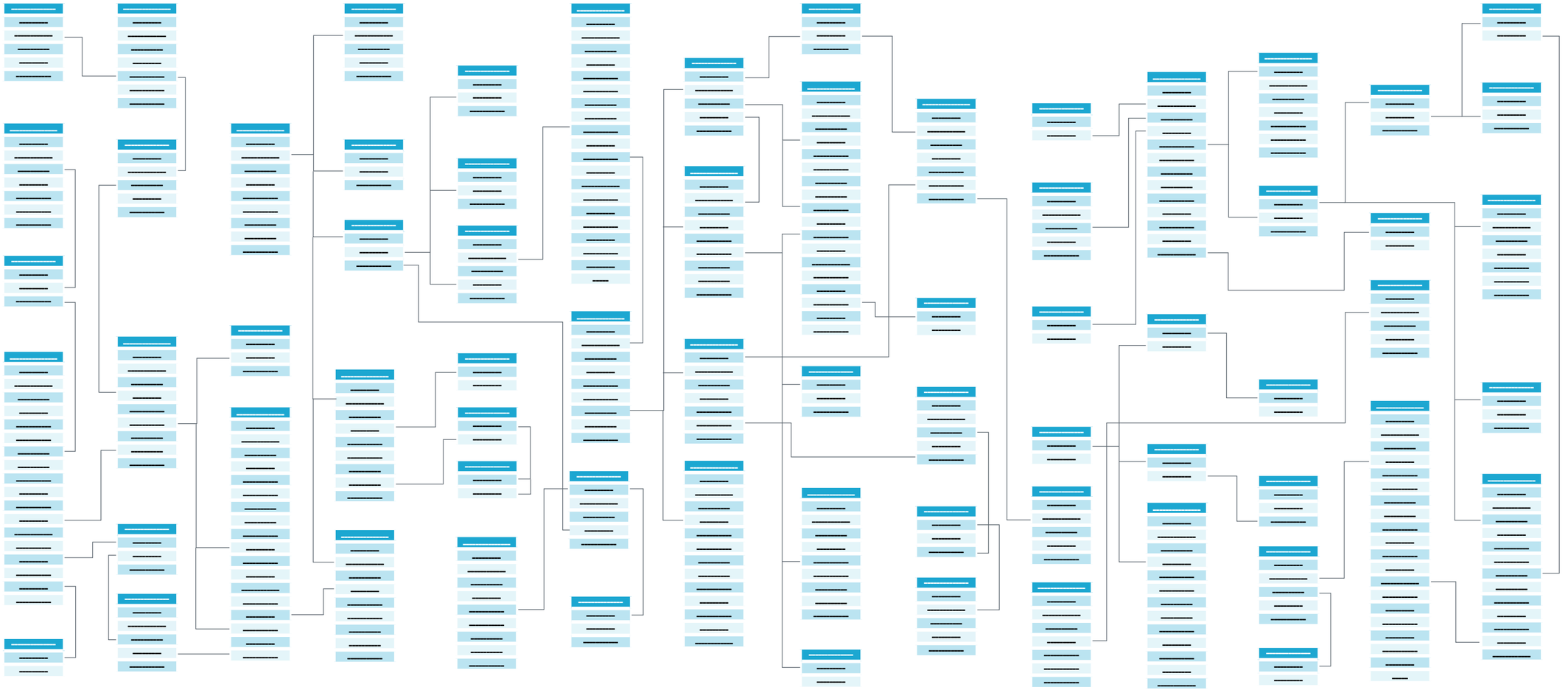
Therein lies the fundamental flaw:
Relational data models are rigid and resistant to change. This hinders business agility.

Modeling the same relationships differently leads to complexity



Unlike Blue Gear Auto Parts, Red Motor Equipment's schema allows for multiple customer phone numbers and addresses. Perhaps this division of Red Motor Equipment processes billing and shipping information and therefore requires both addresses, whereas Blue Gear Auto Parts only processes shipping addresses.

A more realistic looking data schema



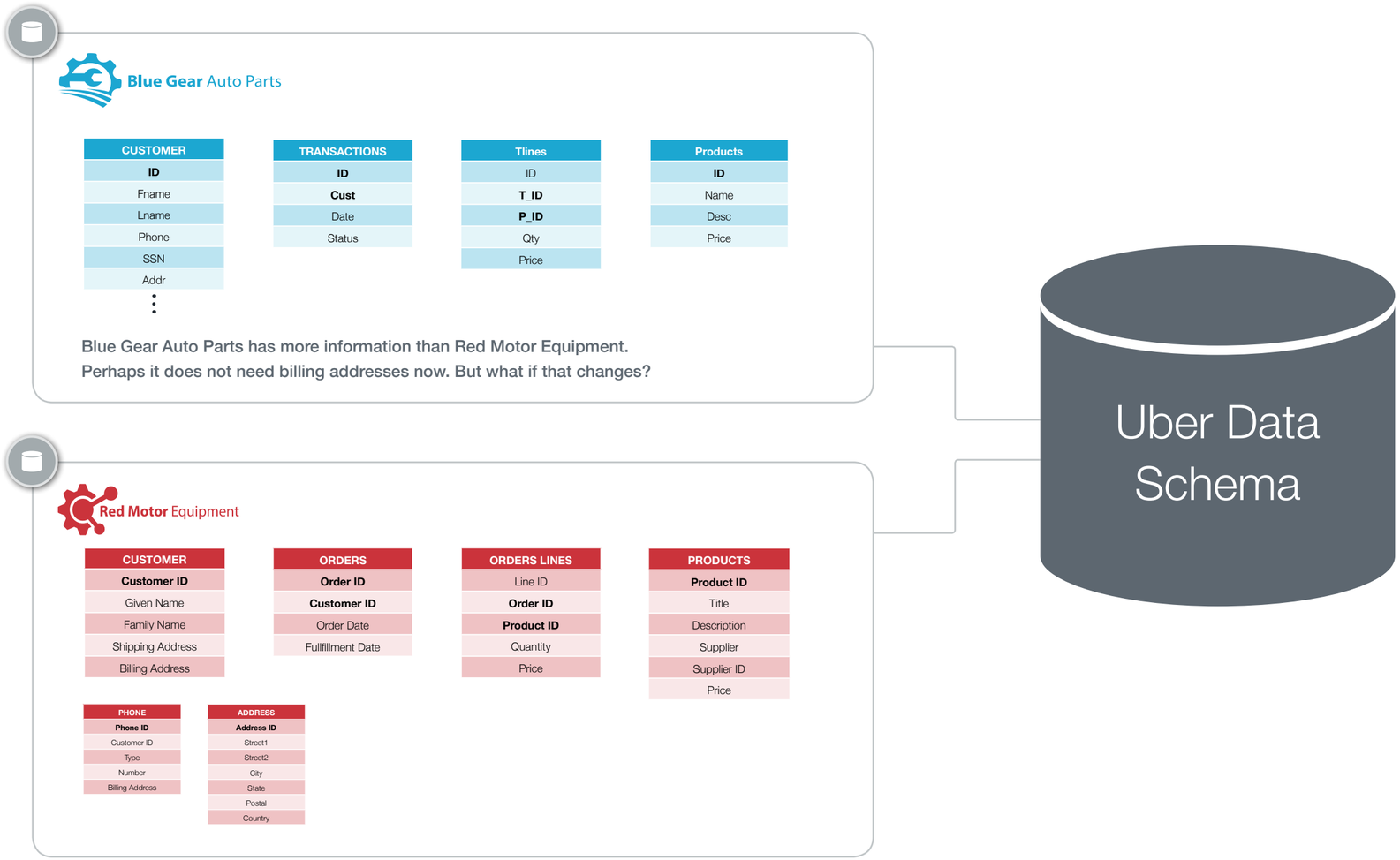
Data schemas are much more complex in the real world. Here is what one simple business application's schema might look like.

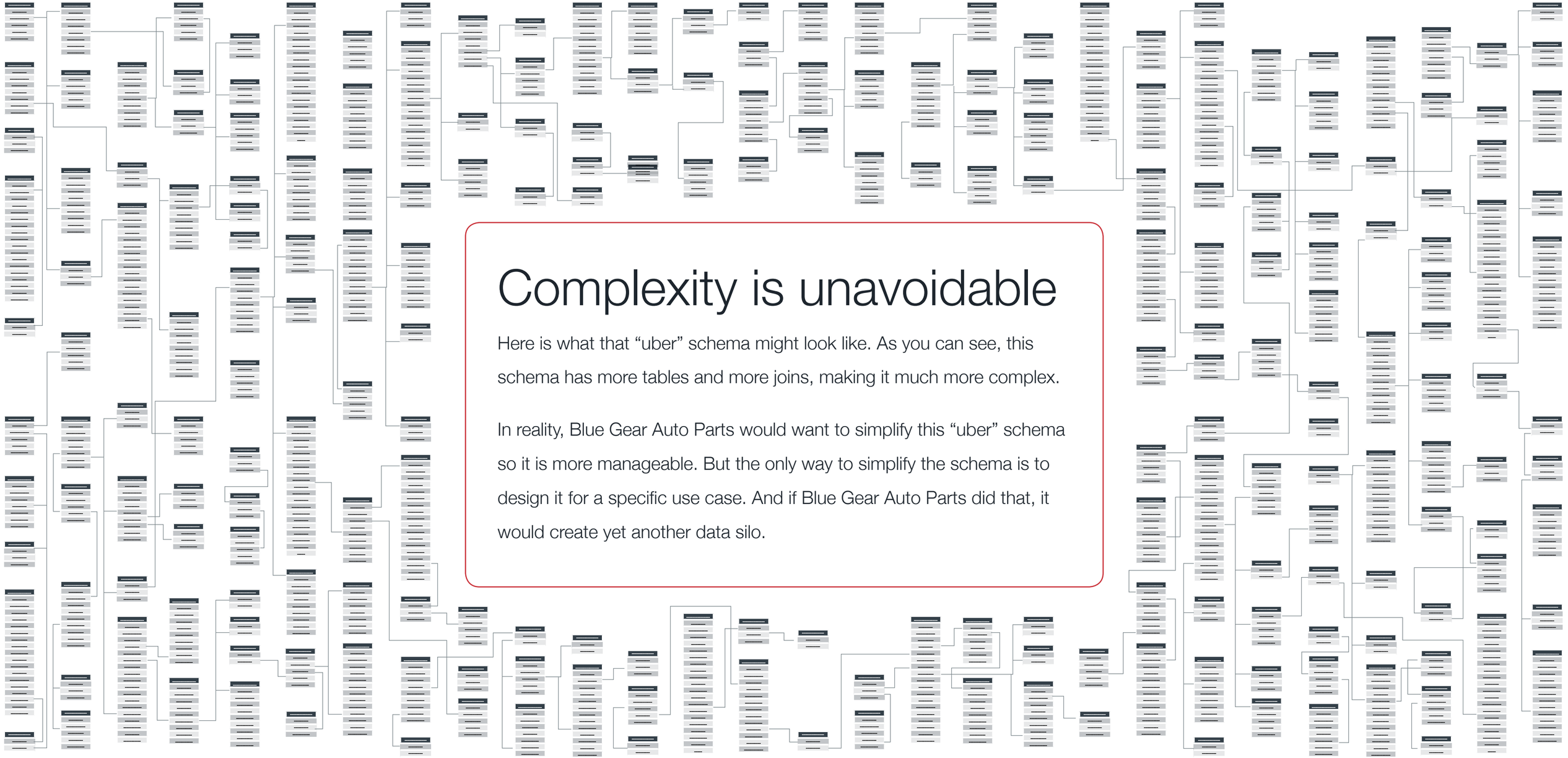
Imagine the complexity of an ERP application schema with tens of thousands of tables.

Integrating schemas creates yet more complexity

Let's say Blue Gear Auto Parts acquires Red Motor Equipment and wants to integrate the two companies' data.

To do that, Blue Gear Auto Parts would need to create a single schema that accommodates all the data from the two companies' source schemas. This will create a more complicated "uber" schema. And in the process, Blue Gear Auto Parts will have to decide what information to keep and what to leave behind.



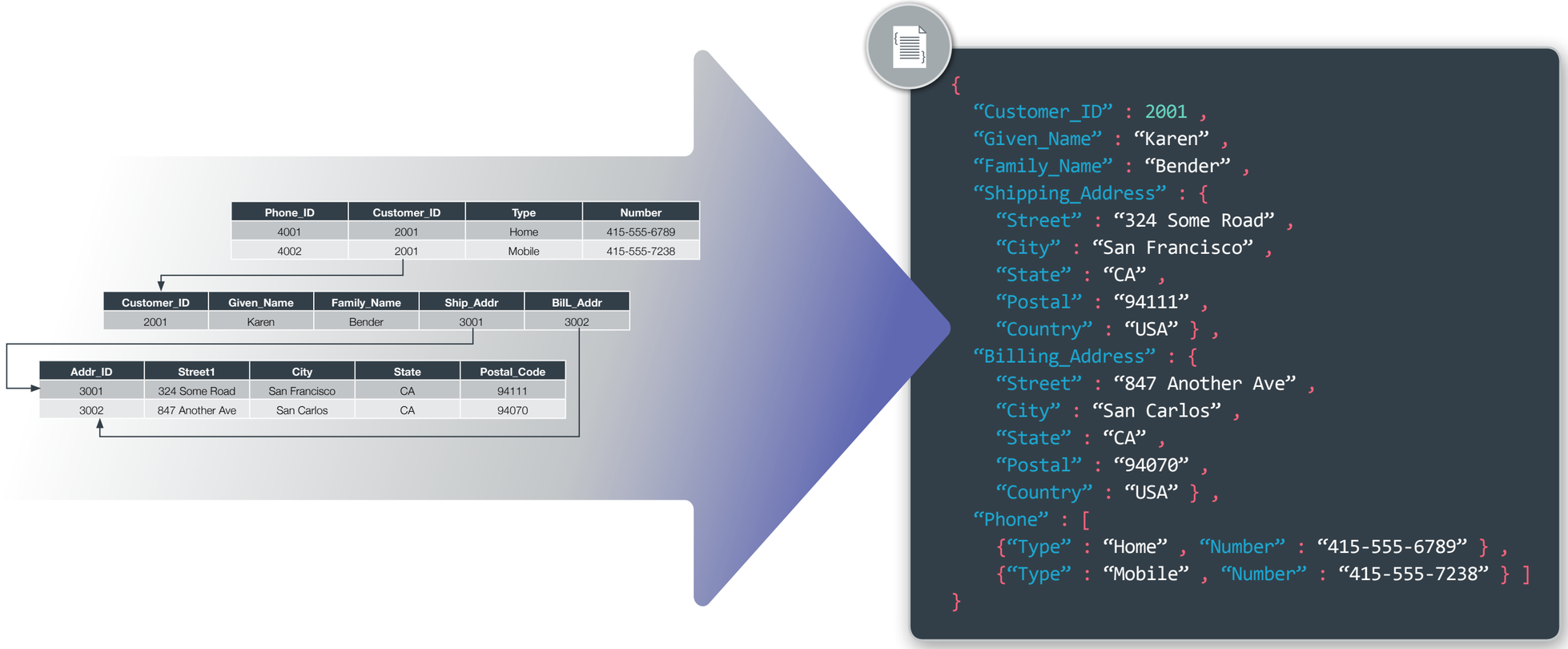


Complexity is unavoidable

Here is what that “uber” schema might look like. As you can see, this schema has more tables and more joins, making it much more complex.

In reality, Blue Gear Auto Parts would want to simplify this “uber” schema so it is more manageable. But the only way to simplify the schema is to design it for a specific use case. And if Blue Gear Auto Parts did that, it would create yet another data silo.

There is a more flexible way to model data



Phone_ID	Customer_ID	Type	Number
4001	2001	Home	415-555-6789
4002	2001	Mobile	415-555-7238

Customer_ID	Given_Name	Family_Name	Ship_Addr	BilL_Addr
2001	Karen	Bender	3001	3002

Addr_ID	Street1	City	State	Postal_Code
3001	324 Some Road	San Francisco	CA	94111
3002	847 Another Ave	San Carlos	CA	94070

Both these records contain the same information. But the customer record on the right shows the hierarchical structure of a document model, which organizes data naturally, like a document. It features arrays (shipping and billing addresses) and nested arrays (phone numbers).

MarkLogic is incredibly flexible

When adding information to MarkLogic®, you simply add the information you have; you do not need to worry about information you do not have. You can represent repeating hierarchical attributes such as phone numbers and addresses naturally, without having to build out separate tables. What's more, your data is immediately queryable, because MarkLogic indexes your data as you add it.

Harmonize data with ease

Rather than using traditional ETL processes that transform data before loading it into a database, MarkLogic's flexible document data model allows you to harmonize the data you need, when you need it. This harmonization happens in a fully transactional database where data can be tracked and managed. With MarkLogic, you can easily pull pieces of data into a canonical model so that data can be queried consistently. Your model can evolve over time as your business needs change. At no point are you required to destroy your raw data.

Know your data lineage

MarkLogic uses an envelope pattern to model data. This allows you to store and query metadata, source data and canonical data all in a single document.

```
{ "metadata" : {
    "Source" : "Finance" ,
    "Date" : "2016-04-17" ,
    "Lineage" : "v01 transform" } ,
  "canonical" : { "Postal" : [ 94111 ] } ,
  "source" : { "ID" : 1001 ,
    "Fname" : "Paul" ,
    "Lname" : "Jackson" ,
    "Phone" : "415-555-1212 | 415-555-1234" ,
    "SSN" : "123-45-6789" ,
    "Addr" : "123 Avenue Road" ,
    "City" : "San Francisco" ,
    "State" : "CA" ,
    "Zip_Code" : 94111 } }
}
```

Metadata is added to the document to preserve data provenance and lineage.

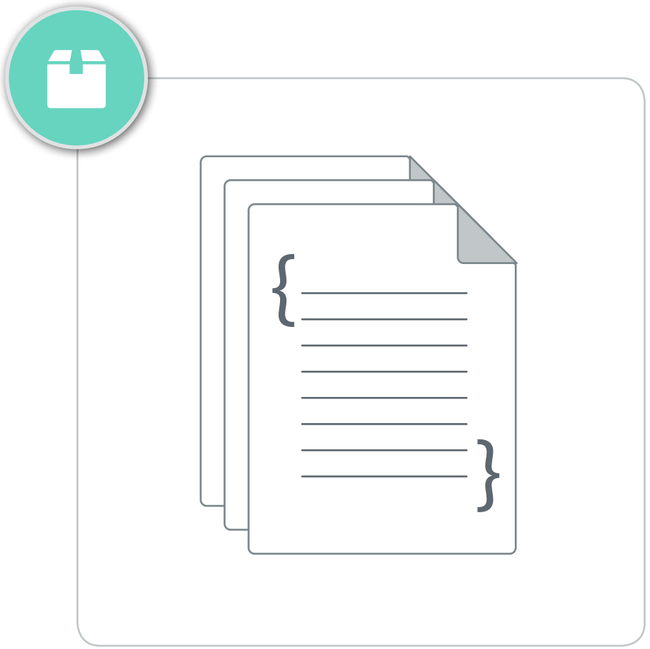
```
{ "metadata" : {
    "Source" : "POS" ,
    "Date" : "2016-04-17" ,
    "Lineage" : "v01 transform" } ,
  "canonical" : { "Postal" : [ 94111 , 94070 ] } ,
  "source" : {   "Customer_ID" : 2001 ,
    "Given_Name" : "Karen" ,
    "Family_Name" : "Bender" ,
    "Shipping_Address" : {
      "Street" : "324 Some Road" ,
      "City" : "San Francisco" ,
      "State" : "CA" ,
      "Postal_Code" : "94111" ,
      "Country" : "USA" } ,
    "Billing_Address" : {
      "Street" : "847 Another Ave" ,
      "City" : "San Carlos" ,
      "State" : "CA" ,
      "Postal_Code" : "94070" ,
          :
```

Entities = Documents

The document data model offers the most flexible and iterative approach for modeling business entities.

Customer

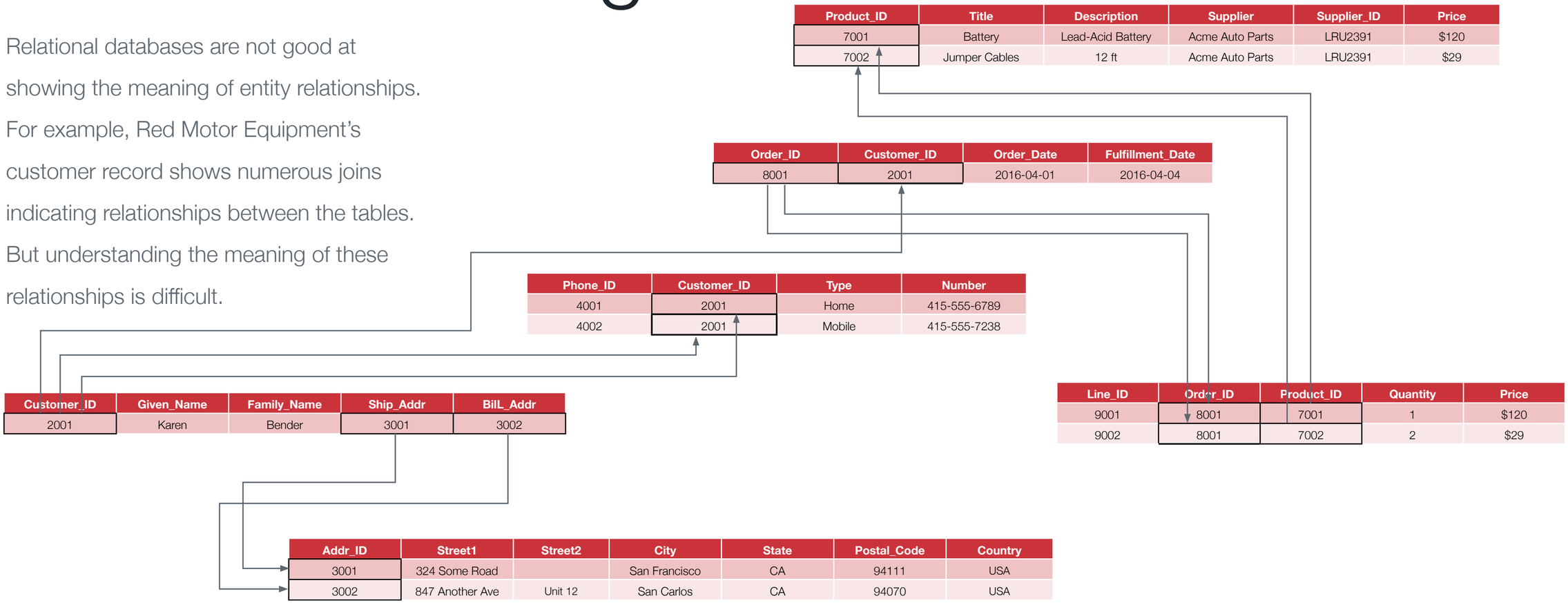
Transaction



Product

Relational databases obscure meaning

Relational databases are not good at showing the meaning of entity relationships. For example, Red Motor Equipment's customer record shows numerous joins indicating relationships between the tables. But understanding the meaning of these relationships is difficult.



Product_ID	Title	Description	Supplier	Supplier_ID	Price
7001	Battery	Lead-Acid Battery	Acme Auto Parts	LRU2391	$120
7002	Jumper Cables	12 ft	Acme Auto Parts	LRU2391	$29

Order_ID	Customer_ID	Order_Date	Fulfillment_Date
8001	2001	2016-04-01	2016-04-04

Phone_ID	Customer_ID	Type	Number
4001	2001	Home	415-555-6789
4002	2001	Mobile	415-555-7238

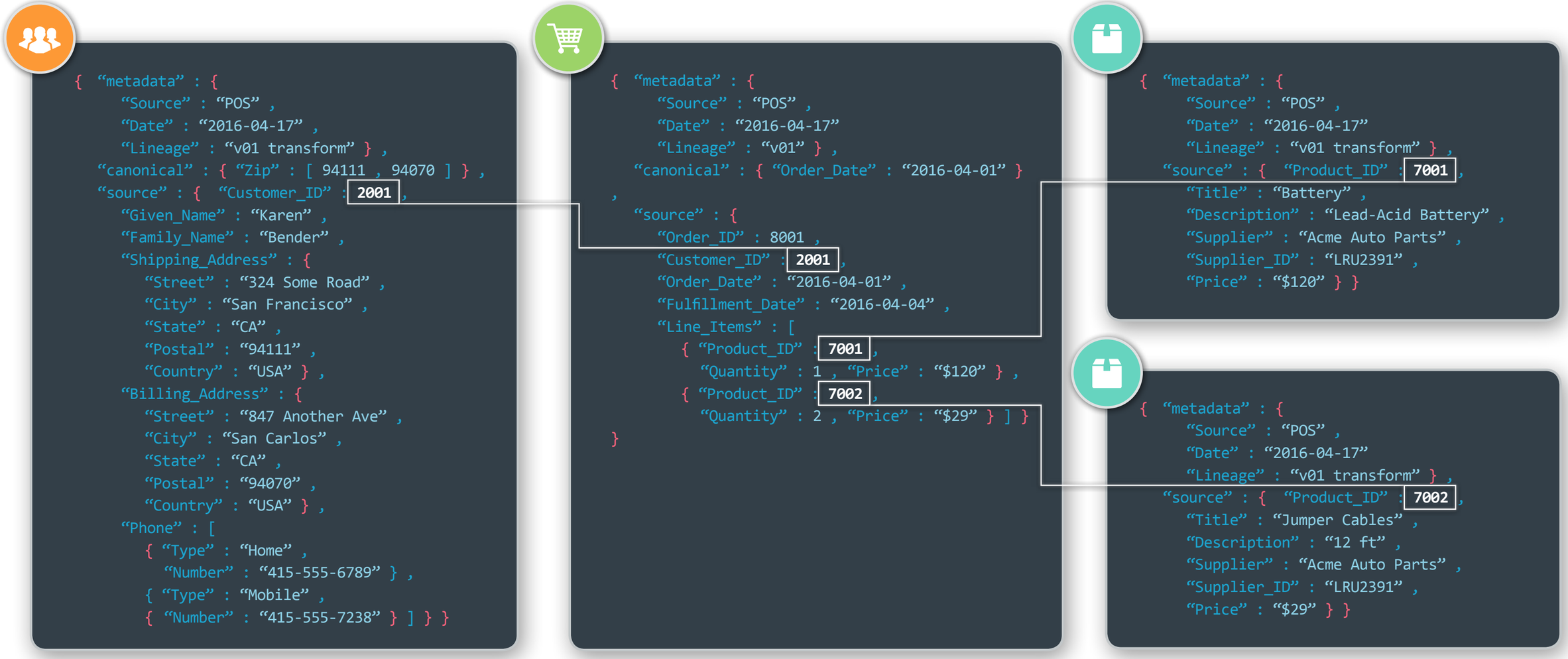
Customer_ID	Given_Name	Family_Name	Ship_Addr	BilL_Addr
2001	Karen	Bender	3001	3002

Line_ID	Order_ID	Product_ID	Quantity	Price
9001	8001	7001	1	$120
9002	8001	7002	2	$29

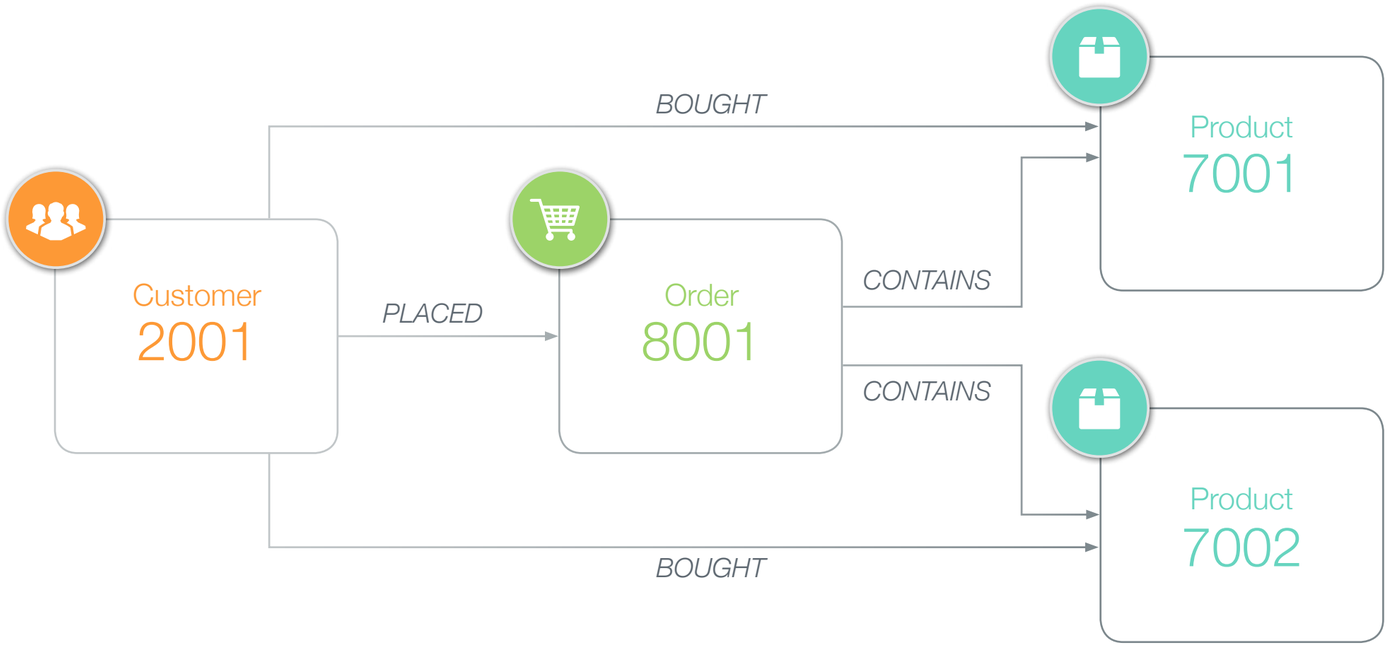
Addr_ID	Street1	Street2	City	State	Postal_Code	Country
3001	324 Some Road		San Francisco	CA	94111	USA
3002	847 Another Ave	Unit 12	San Carlos	CA	94070	USA

Entity relationships in MarkLogic

Here is a view of the same customer record in MarkLogic. You can clearly see that the customer completed a transaction that included a “Battery” and “Jumper Cables.” The entity relationships are much easier to understand.



Discover more with Semantics



SEARCH and DISCOVER:

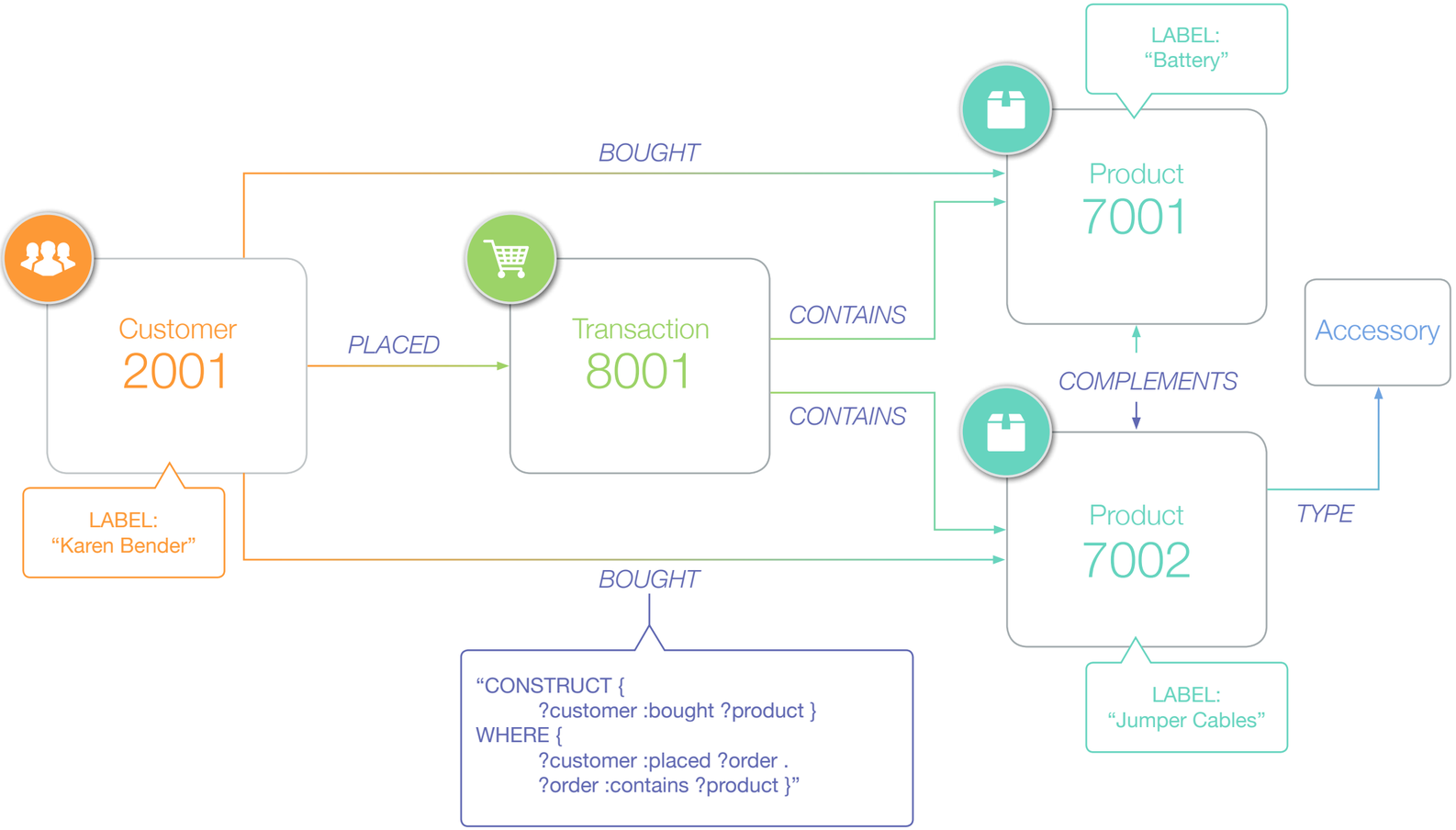
```
SELECT ?customer
WHERE {
  ?customer :placed ?order .
  ?order :contains :Product7002 }

CONSTRUCT { ?customer :bought ?product }
WHERE {
  ?customer :placed ?order .
  ?order :contains ?product }
```

Relational data models require that you understand your schema in order to query your data. That is not the case with MarkLogic. With the power of semantics, MarkLogic makes entity relationships simple to understand, so it is easier and faster to query and discover data. Semantic relationships in MarkLogic are rich with meaning: You can define relationships explicitly and query the meaning of relationships directly. You can also draw inferences from the meaning of relationships to create new information.

Triples form relationships between concepts

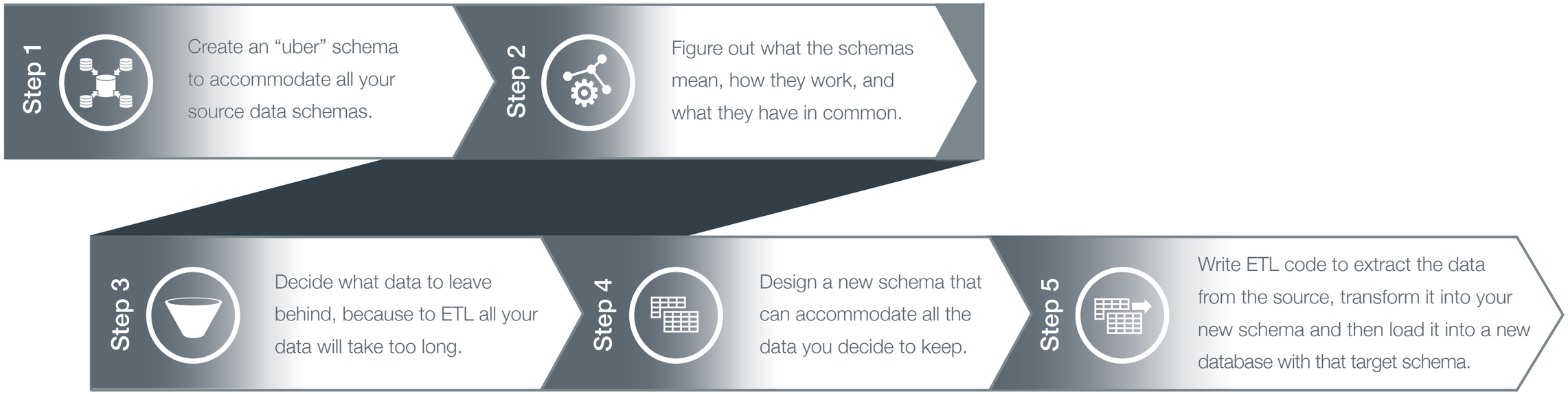
Example: *Karen Bender*, customer 2001, bought product 7001, *a Battery*.

Using triples, Semantics provides the best way to model relationships in your data. Triples consist of a *SUBJECT*, *PREDICATE* and *OBJECT*. They eliminate the need for foreign keys, nested queries, and complex joins. When you combine a document model with Semantics, you get a multi-model approach for all your data.



Relational vs MarkLogic

In a relational database:

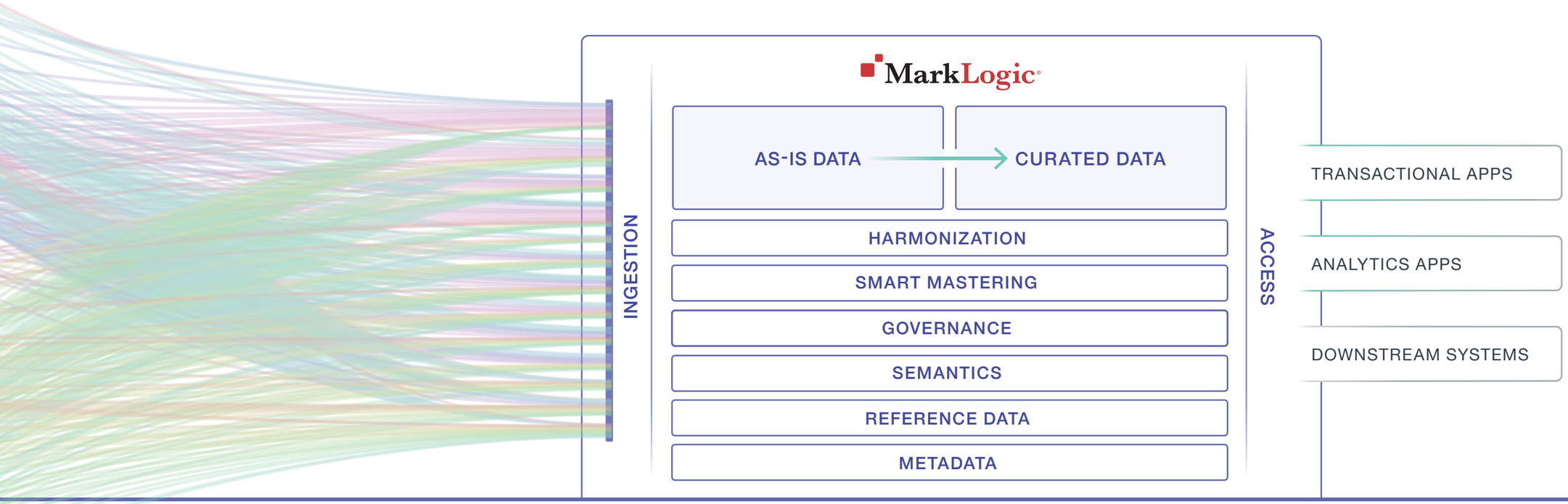


In MarkLogic:

“I’m a strong relational guy, but I fell in love with MarkLogic when I saw it. We could bring everything in, in a schema-agnostic way, and not have to rip and replace.”

- **Alan Campbell**, Software Systems Engineer, Autoliv

Enterprise ready. Unparalleled flexibility.



The MarkLogic Operational Data Hub is an enterprise architecture pattern designed to ingest data as is from any source. It stores diverse data types, including JSON, XML, text, geospatial, and Semantics triples with indexing for fast search. The Operational Data Hub supports ACID transactions and features the scalability, high availability and disaster recovery capabilities necessary for the most demanding environments. And with APIs for fast data access and application development, changing data will no longer hold your business back.

Leading organizations improving business agility with MarkLogic

FINANCIAL SERVICES	J.P.Morgan, ABN·AMRO, Morgan Stanley, Northern Trust, Deutsche Bank, Bank of America Merrill Lynch, Broadridge, citi
INSURANCE	Erie Insurance, The Hanover Insurance Group, hannover re, de Goudse verzekeringen, $100B COMPANY, $100B COMPANY, $10B COMPANY, $5B COMPANY
HEALTHCARE	HealthCare.gov, WellCare Health Plans, Johnson & Johnson, AMGEN, abbvie, HSBlox, L.A. Care Health Plan, AMA American Medical Association
PUBLIC SECTOR	DISA, Ministry of Housing, Communities & Local Government, WORLD BANK GROUP, OECD Better Policies for Better Lives
PUBLISHING	Springer, IHS Markit, Haufe. Group, Columbia University Press, bsi., The New England Journal of Medicine, Elsevier, Wolters Kluwer When you have to be right, THOMSON REUTERS
ENTERTAINMENT	BBC, NBC, DOW JONES, EA Electronic Arts, COMCAST, CONDÉ NAST, SONY
OTHER INDUSTRIES	BOEING, ENSCO, CISCO, salesforce, Autoliv, EATON, DHL, Dow

MarkLogic's flexible data model allows you to integrate data faster and more cost effectively.

Free your data with MarkLogic.

Learn more about MarkLogic solutions

Or download a free version of MarkLogic