Microservices
Q/A

Microservice Q/A

1. Database design

Database load ?

Reads

Data volume

Writes

Optimize reads

Problems with query data

Complex query, join, aggregation

Large scans

Misalignment schema and query

Optimize reads

Problems with query data

- Complex query, join, aggregation
- Large scans
- Misalignment schema and query

Complex query !!

SELECT COUNT(*) ... GROUP BY USER_ID

Transaction	
ID	USER_ID
1	U001
2	U001
3	U002
4	U002

Complex query !!

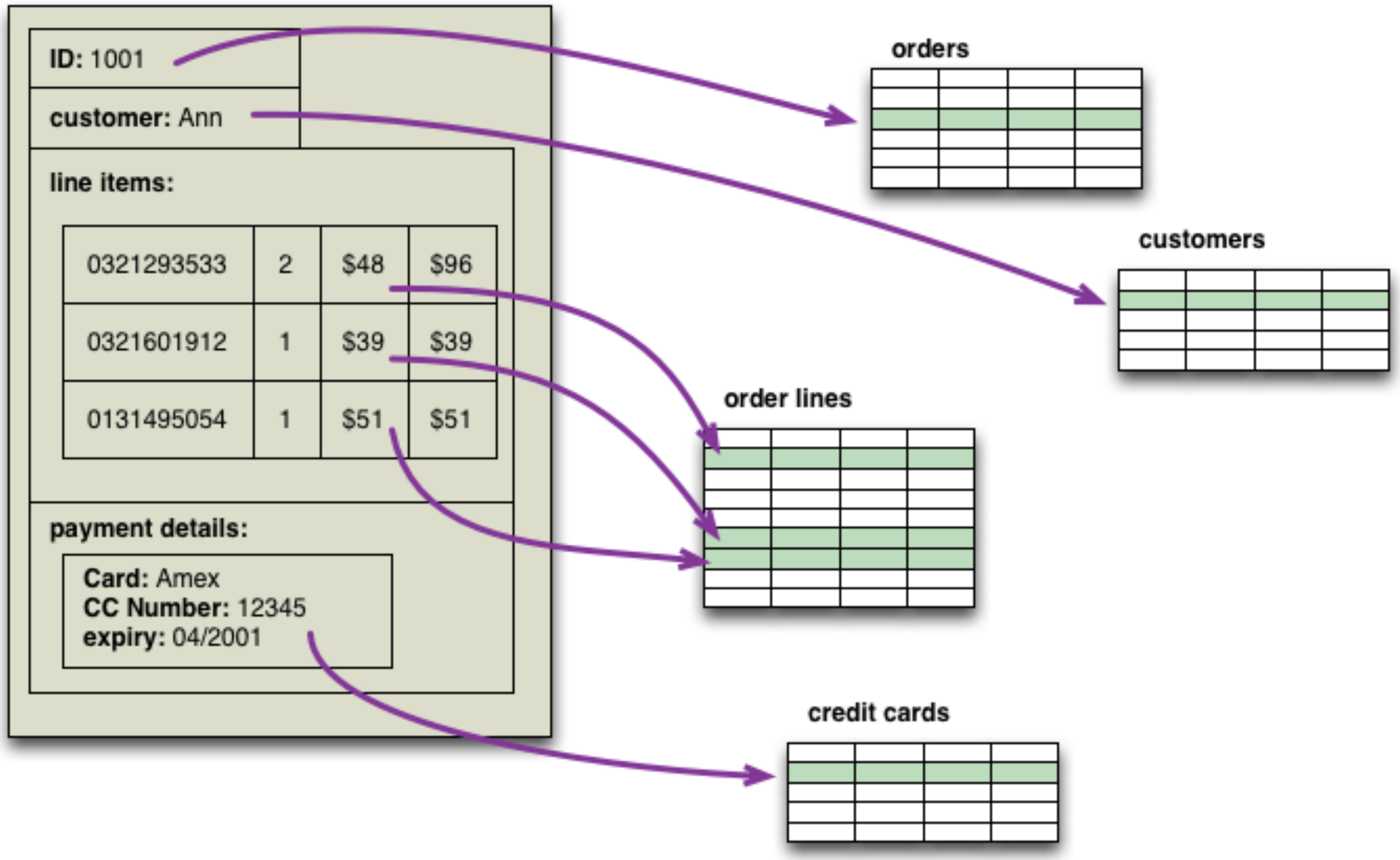
SELECT COUNT(*) ... GROUP BY USER_ID

Transaction	
ID	USER_ID
1	U001
2	U001
3	U002
4	U002

Transaction_count	
USER_ID	COUNT
U001	2
U001	2

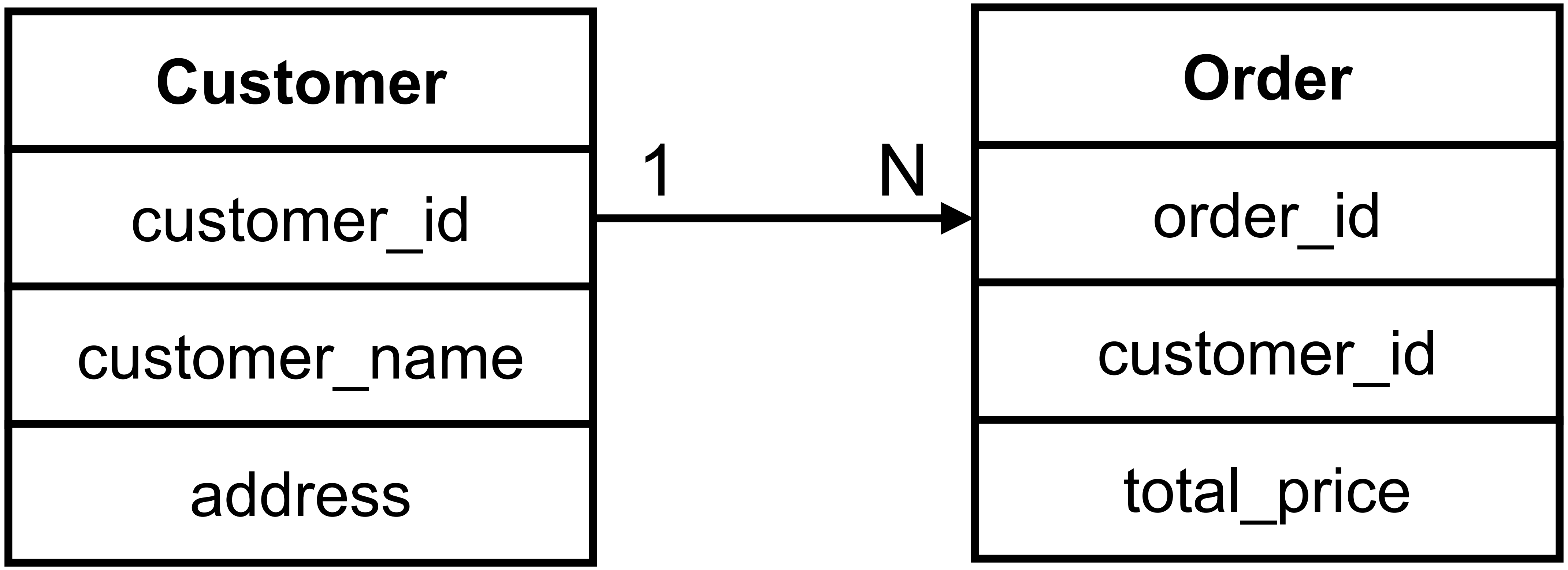
Complex query !!

Query and JOINs



https://martinfowler.com/bliki/AggregateOrientedDatabase.html

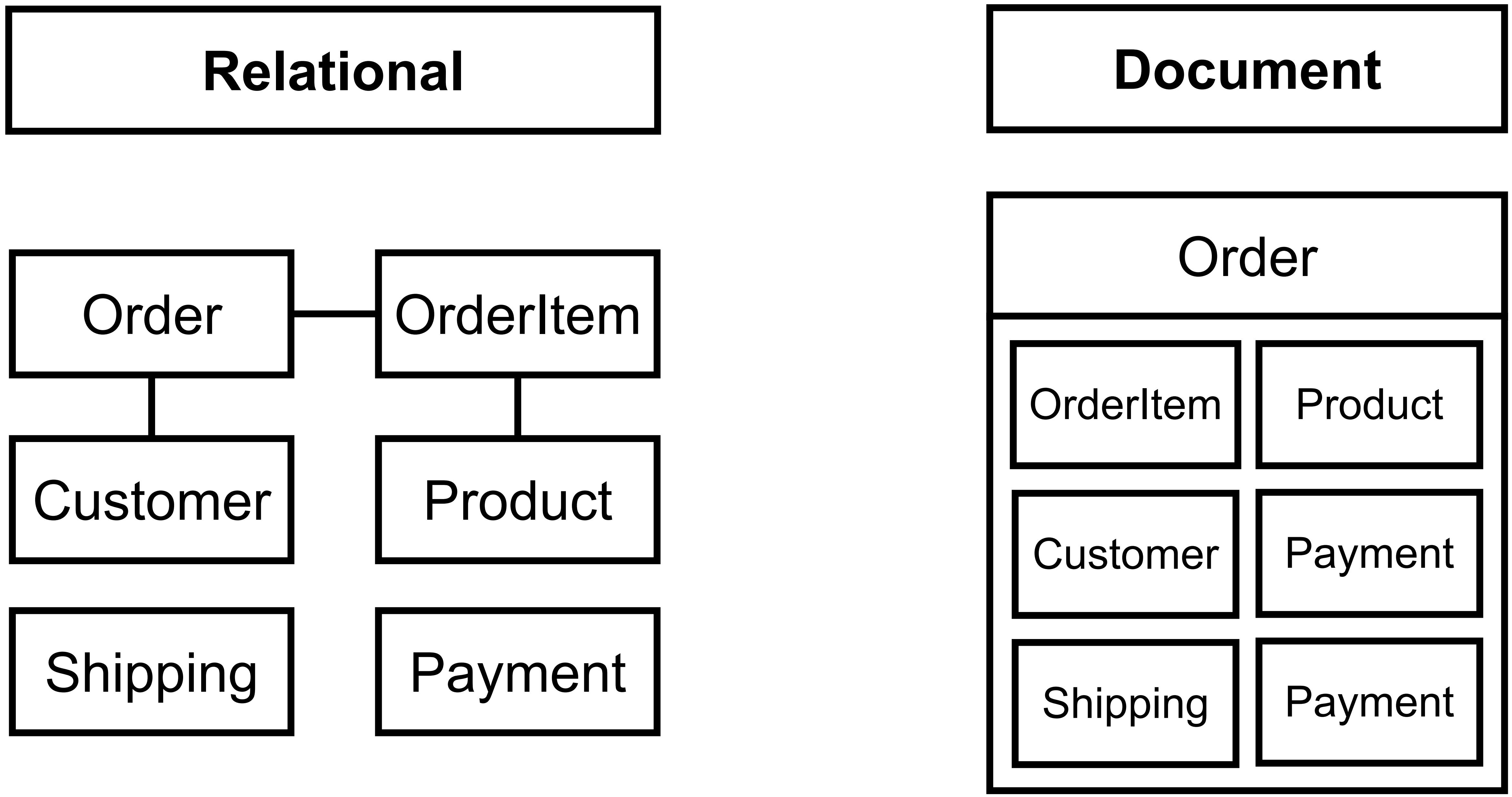
Joins


Customer
customer_id
customer_name
address
1
N
Order
order_id
customer_id
total_price

Pre-joins (Redundant column)

Order
order_id
customer_id
total_price
customer_name

SQL vs NoSQL


Relational
Order
OrderItem
Customer
Product
Shipping
Payment
Document
Order
OrderItem
Product
Customer
Payment
Shipping
Payment

Optimize reads

Problems with query data

Complex query, join, aggregation

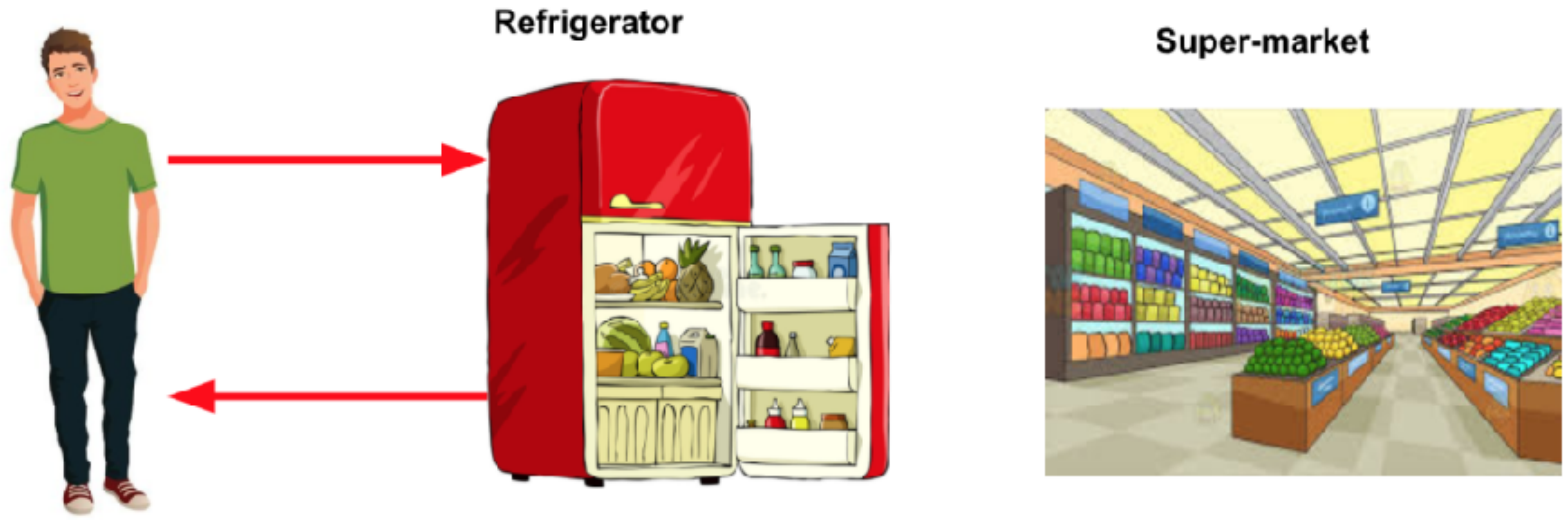
Large scans

Misalignment schema and query

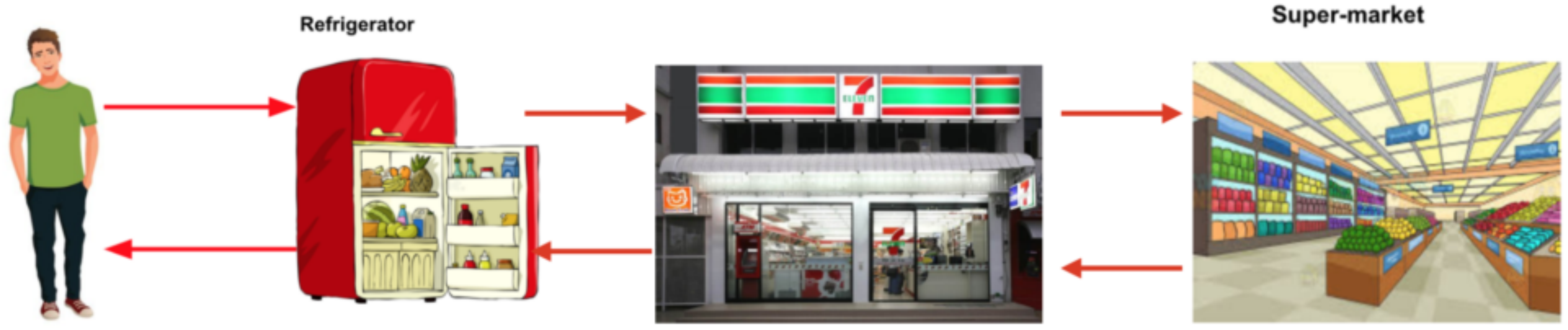
Large number of rows

LIMIT
Pagination

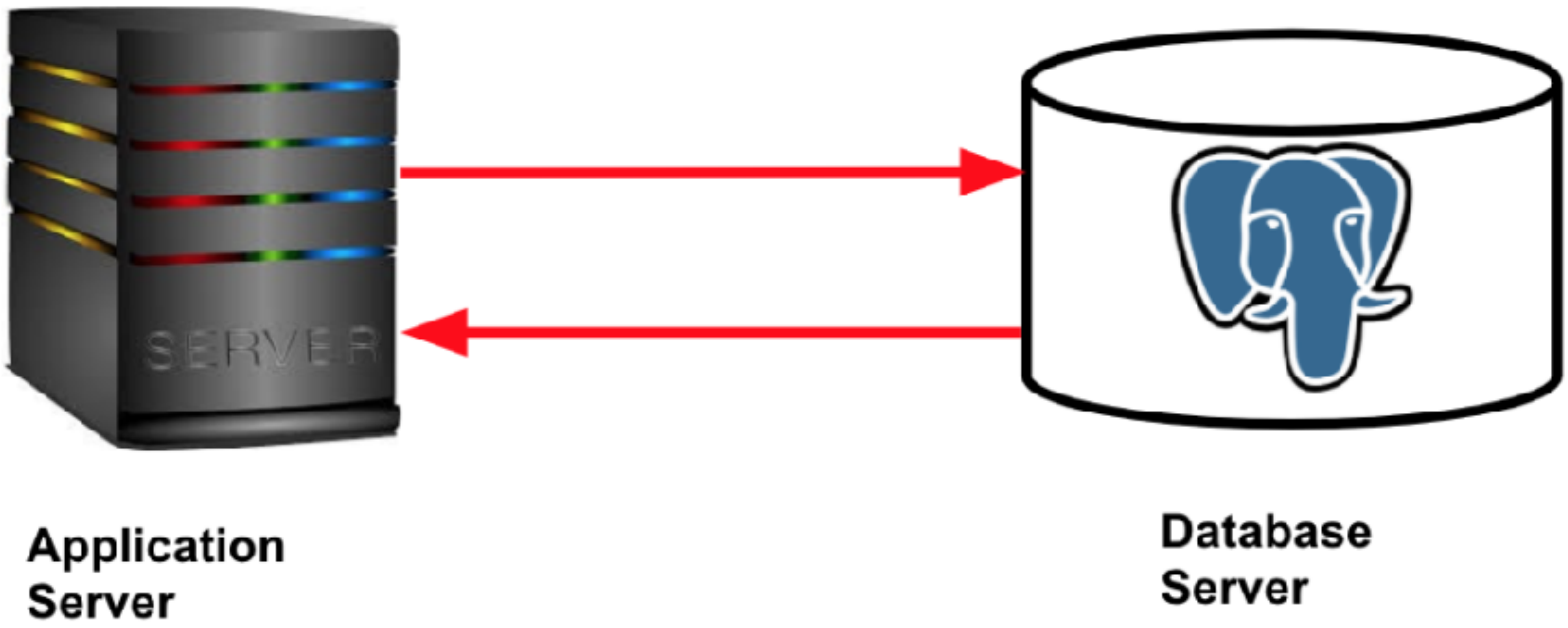
Caching Data



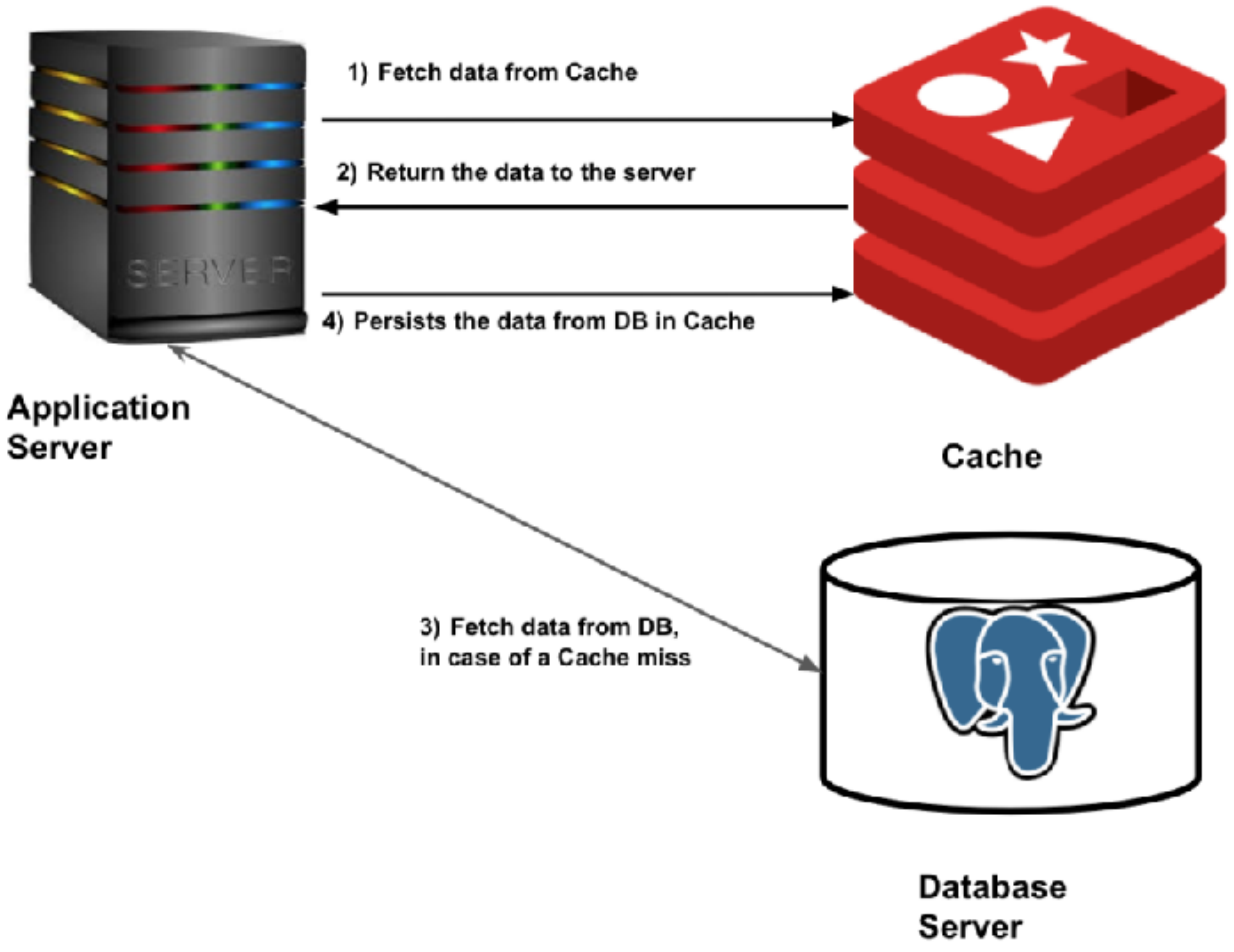
Caching Data



Caching Data



Caching Data


1) Fetch data from Cache
2) Return the data to the server
4) Persists the data from DB in Cache
SERVER
Application Server
Cache
3) Fetch data from DB, in case of a Cache miss
Database Server

2. gRPC

https://grpc.io/

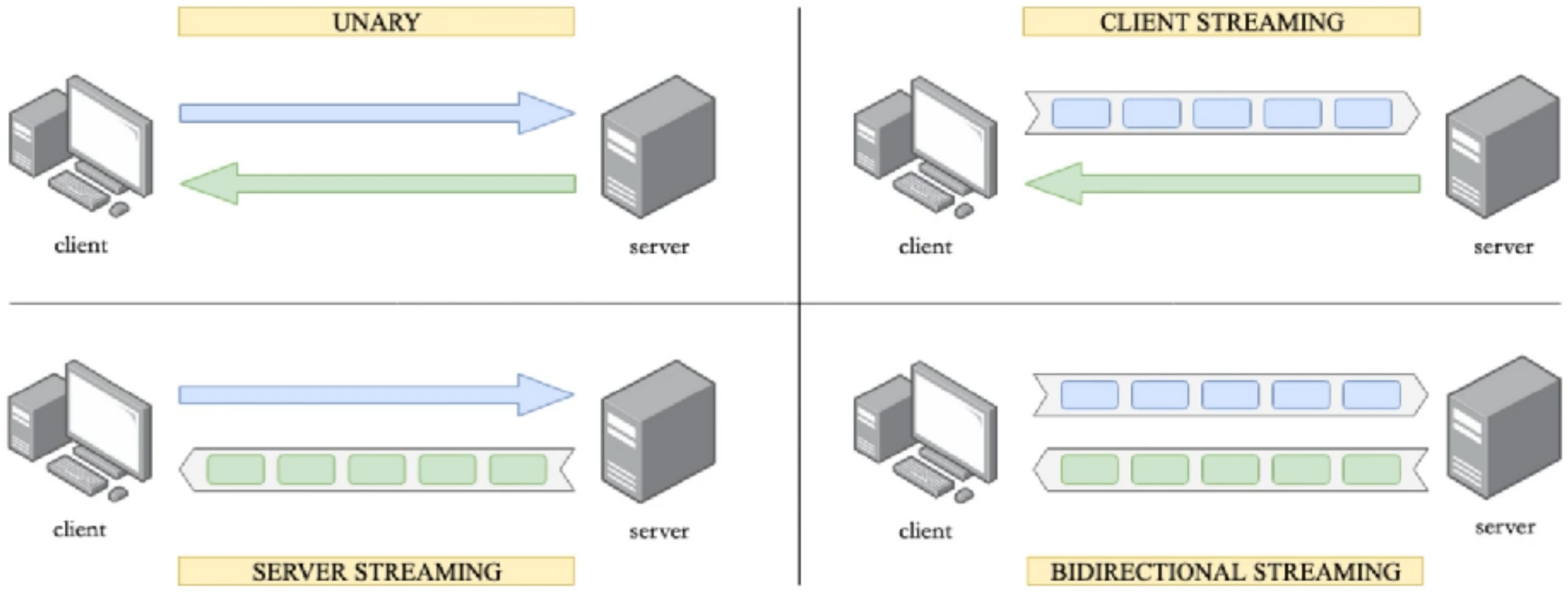
gRPC vs REST

gRPC uses HTTP/2
REST uses HTTP 1

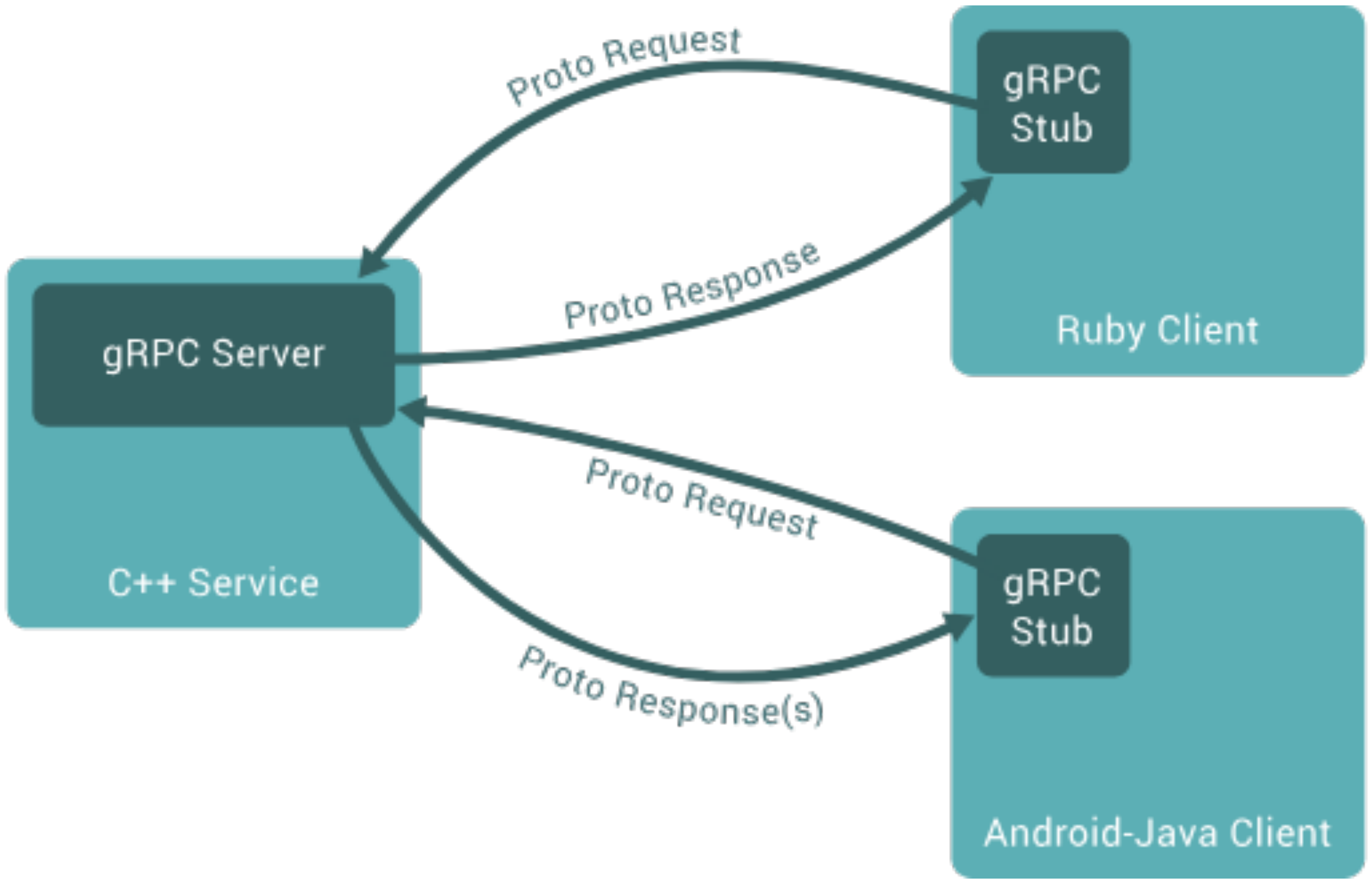
gRPC vs REST

Properties	gRPC	REST
HTTP	HTTP 2	HTTP 1.1
Message format	Protobuf (less size)	JSON/XML (more size)
Communication	Client-request, bidirectional, streaming	Client-request only
Implementation time	More	Less
Code generation	Native Protoc compiler	3-party library Swagger

HTTP 2

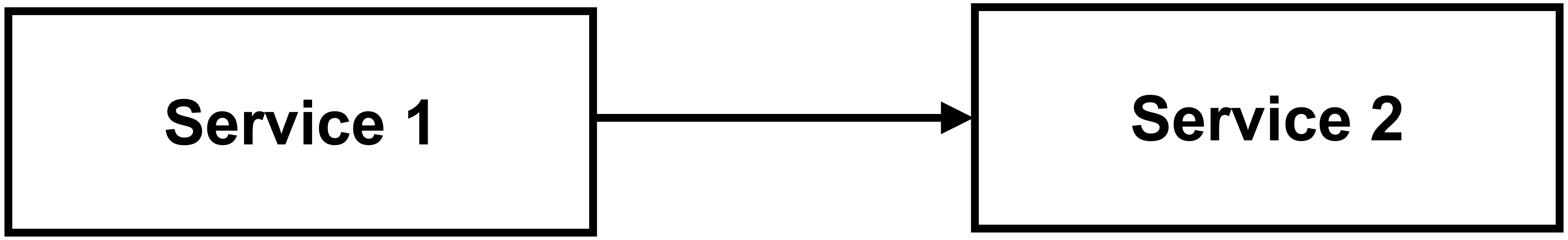

UNARY
client
server
CLIENT STREAMING
client
server
client
server
SERVER STREAMING
client
server
BIDIRECTIONAL STREAMING

gRPC implementation



https://grpc.io/docs/what-is-grpc/introduction/

gRPC implementation



Step 1 :: Design a protobuf format and generate code

Protobuf format

Protocol buffers

```
message Person {
  optional string name = 1;
  optional int32 id = 2;
  optional string email = 3;
}
```

A proto definition.

```
// Java code
Person john = Person.newBuilder()
    .setId(1234)
    .setName("John Doe")
    .setEmail("jdoe@example.com")
    .build();
output = new FileOutputStream(args[0]);
john.writeTo(output);
```

Using a generated class to persist data.

```
// C++ code
Person john;
fstream input(argv[1],
    ios::in | ios::binary);
john.ParseFromIstream(&input);
id = john.id();
name = john.name();
email = john.email();
```

Using a generated class to parse persisted data.

https://protobuf.dev/