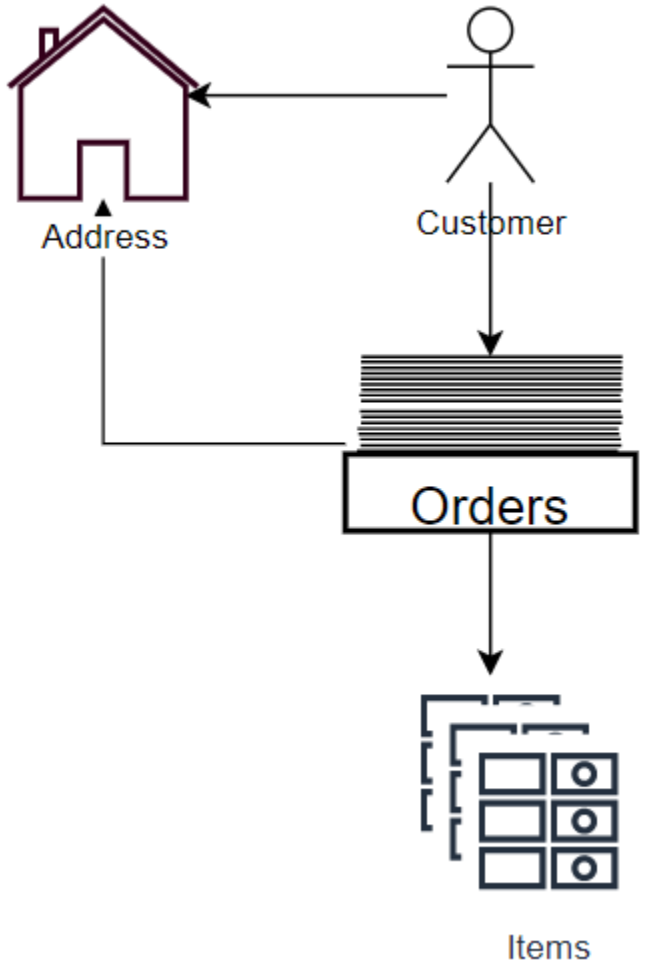
BeSA
Become a Solutions Architect

Data modelling by example

Amazon RDS

Amazon DynamoDB

Example



Customers have many delivery addresses

Customers have many orders

Orders have a single delivery address

Orders have many items

RDS

Order#	Address	Item	Email	Name
ORD001	1 BeSA Lane	Pizza	james@besa.com	James Eastham
ORD001	1 BeSA Lane	Fries	james@besa.com	James Eastham
ORD001	1 BeSA Lane	Soft Drink	james@besa.com	James Eastham
	1 BeSA Lane		james@besa.com	James Eastham
			ashish@besa.com	Ashish Prajapati

RDS

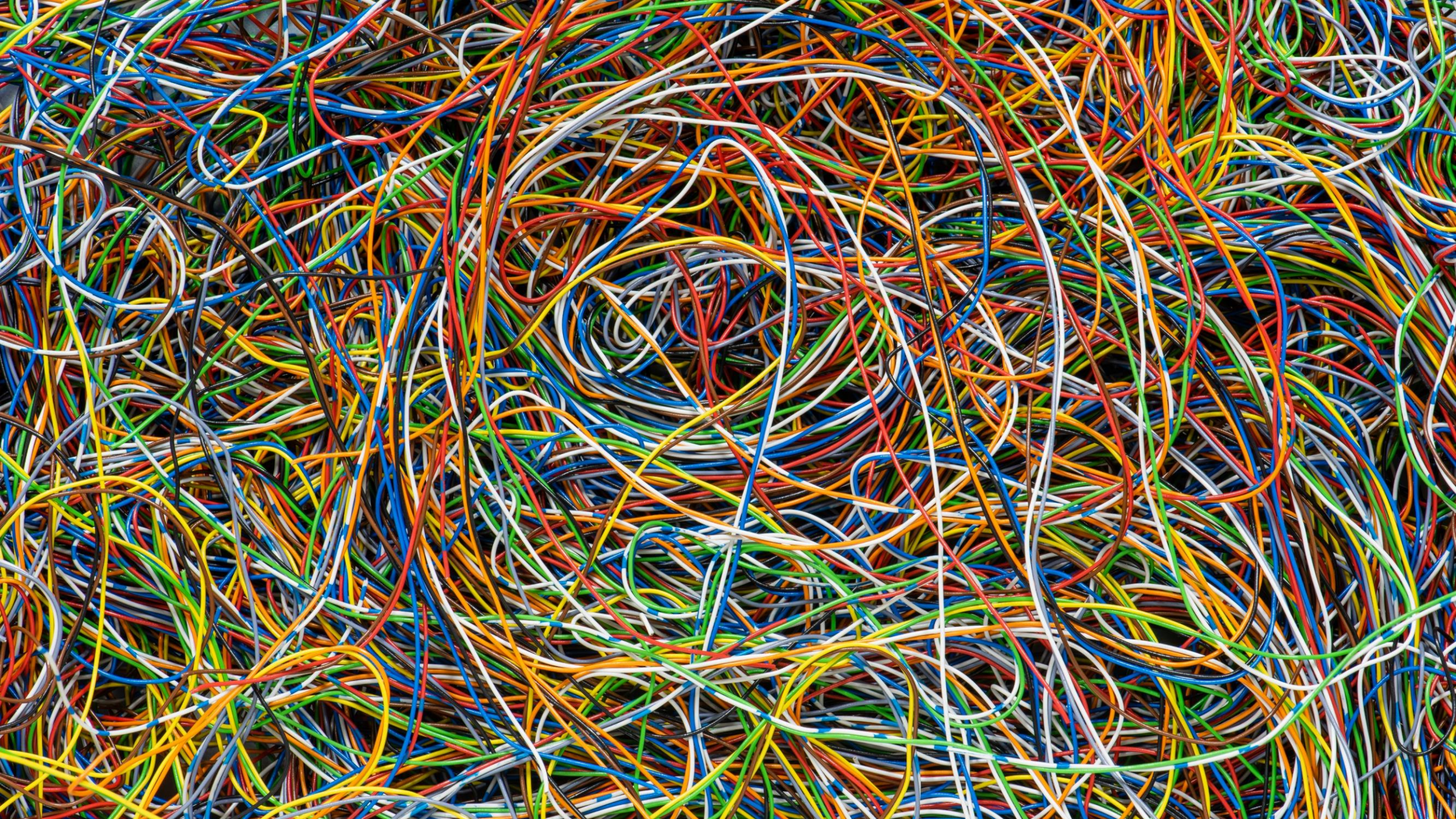
- Start with the **model**
- **Normalize** your data
- **Minimize** data duplication and storage

CustomerId	Email	Name
1	james@besa.com	James Eastham
2	ashish@besa.com	Ashish Prajapati

AddressId	CustomerId	Address
1	1	1 BeSA lane

OrderId	CustomerId	AddressId	Order#
50	1	1	ORD001

OrderItemId	OrderId	Item
1	50	Pizza
2	50	Fries
3	50	Soft Drink



Start with your application access patterns

- Retrieve a specific customer
- List all customer orders
- List all customer addresses
- Retrieve a specific order
- Retrieve the items for an order

NoSQL

- Retrieve a specific customer:
  PK = CUSTOMER#JAMES@BESA.COM
  SK = #CUSTOMERDATA
- List all customer addresses:
  PK = CUSTOMER#JAMES@BESA.COM
  SK starts_with ADDRESS
- Retrieve a specific order:
  PK = ORD#ORD001
  SK = ORDER#ORD001
- Retrieve the items for an order:
  PK = ORD#ORD001
  SK starts_with ORDERITEM

Primary key		Attributes			
Partition key: PK	Sort key: SK				
CUSTOMER#JAMES@BESA.COM	#CUSTOMERDATA	CustomerName	Type	EmailAddress	
		James Eastham	Customer	james@besa.com	
	ADDRESS#001	Address	Type		
		1 BeSA Lane	Address		
ORDER#ORD001	ORDER#ORD001	OrderNumber	Address	GSI1PK	Type
		ORD001	1 BeSA Lane	CUSTOMER#JAMES@BESA.COM	Order
	ORDERITEM#001	Item	Type		
		Pizza	OrderItem		
	ORDERITEM#002	Item	Type		
		Fries	OrderItem		
	ORDERITEM#003	Item	Type		
		Soft Drink	OrderItem		
CUSTOMER#ASHISH@BESA.COM	#CUSTOMERDATA	CustomerName	Type	EmailAddress	
		Ashish Prajapati	Customer	ashish@besa.com	
CUSTOMER#PRASAD@BESA.COM	#CUSTOMERDATA	CustomerName	Type	EmailAddress	
		Prasad Rao	Customer	prasad@besa.com	
	ADDRESS#001	Address	Type		
		10 Cloud Road	Address		

NoSQL

- List all customer orders
  Global Secondary Index
  GSI1PK = CUSTOMER#JAMES@BESA.COM

Primary key	Attributes				
Partition key: GSI1PK					
CUSTOMER#JAMES@BESA.COM	PK	SK	OrderNumber	Address	Type
	ORDER#ORD001	ORDER#ORD001	ORD001	1 BeSA Lane	Order

Ok, so which would do I choose?

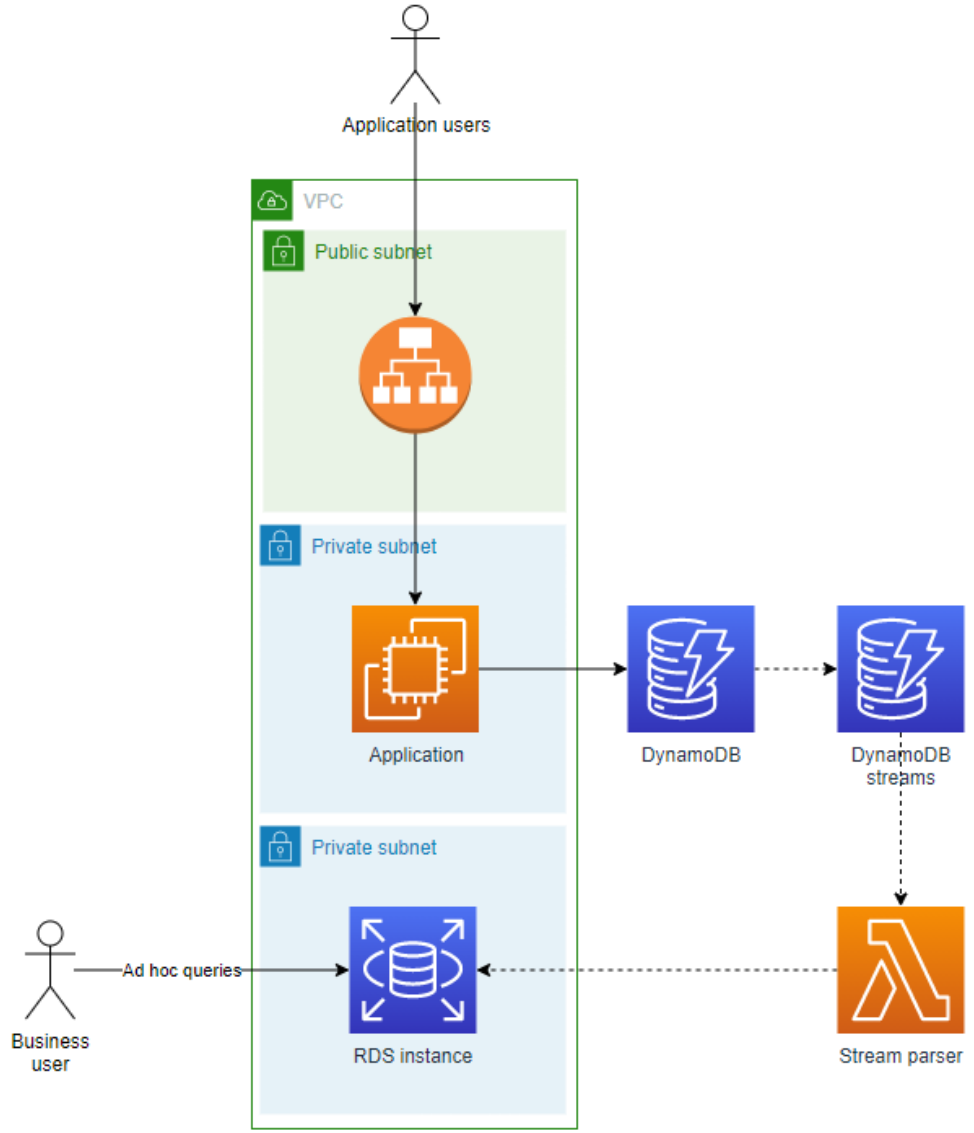
Choose SQL when…

- You have highly structured data that does not change often
- You need transactional integrity (ACID) e.g. accounting or financial use cases
- You often perform complex and ad-hoc queries
- You don’t require horizontal scaling

Choose NoSQL when…

- You have large amounts of un-structured data
- You have a dynamic schema that may change overtime
- You require a database that can easily scale horizontally, possible over multiple regions
- You don’t require the data integrity offered by SQL databases (BASE)

The two can work together



DynamoDB Resources

- #1 resource on DynamoDB data modelling https://www.dynamodbbook.com/
- A decision framework for helping to choose a database https://www.alexdebrie.com/posts/choosing-a-database-with-pie/
- AWS DynamoDB best practices https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/best-practices.html