Databases on AWS: How To Choose The Right Database

Randall Hunt
Software Engineer at AWS
@jrhunt
randhunt@amazon

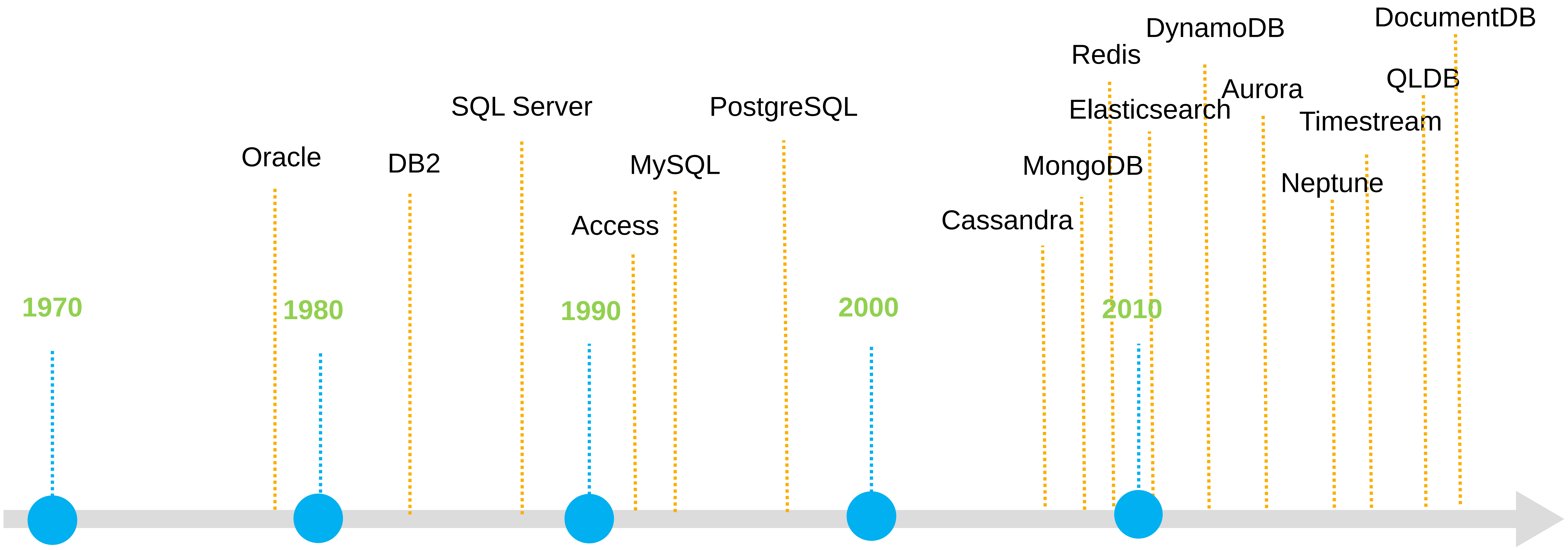
Markus Ostertag
CEO at Team Internet AG
@Osterjour

Dr. Sebastian Brandt
Senior Key Expert - Knowledge Graph at Siemens CT

aws SUMMIT

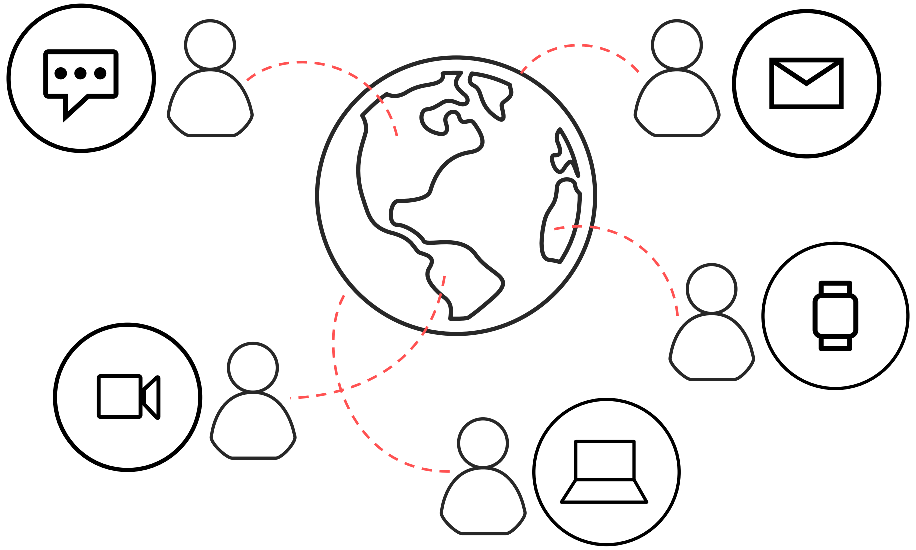
A Quick History of Databases

aws SUMMIT



DocumentDB
DynamoDB
Redis
QLDB
Aurora
SQL Server
PostgreSQL
Elasticsearch
Timestream
Oracle
DB2
MySQL
MongoDB
Neptune
Access
Cassandra
1970
1980
1990
2000
2010

Modern apps have modern requirements



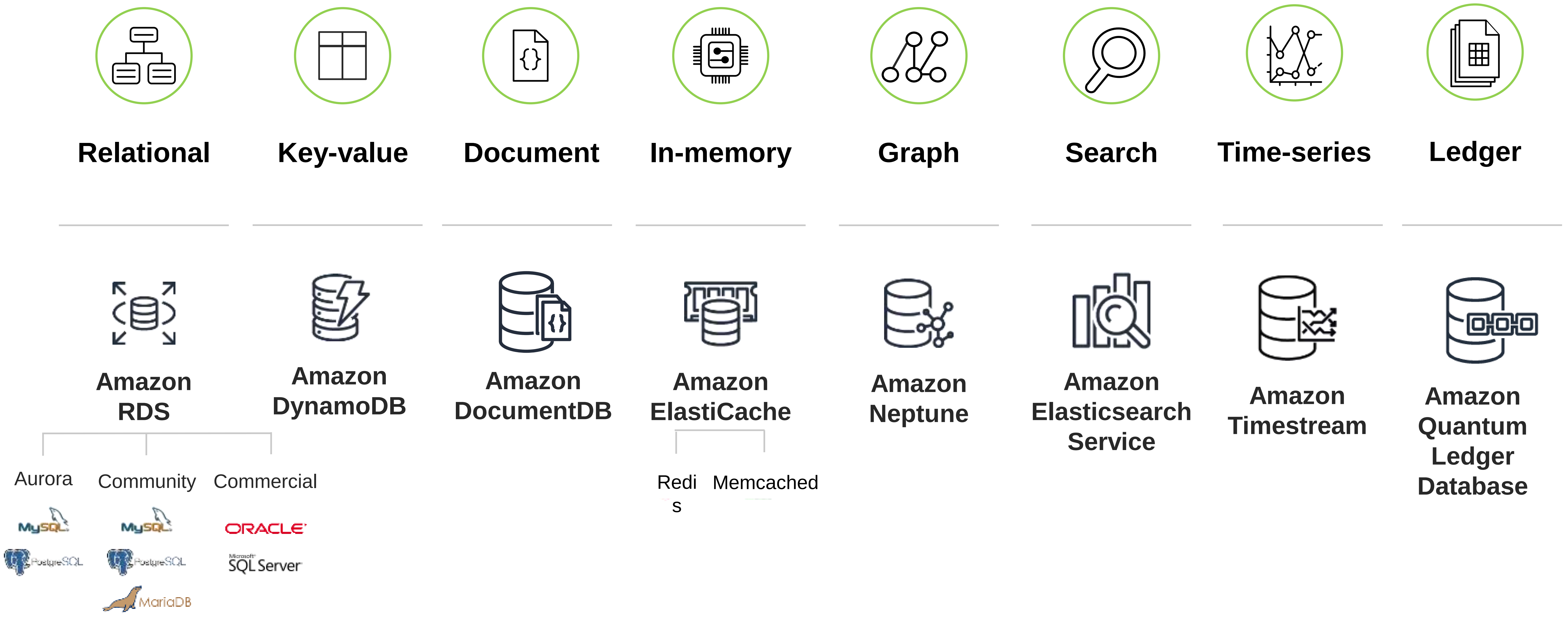
Ride hailing **Media streaming** **Social media** **Dating**

Users:	1 million+
Data volume:	TB–PB–EB
Locality:	Global
Performance:	Milliseconds–microseconds
Request rate:	Millions per second
Access:	Web, Mobile, IoT, Devices
Scale:	Up-down, Out-in
Economics:	Pay for what you use
Developer access:	No assembly required

Common data categories and use cases

Relational	Key-value	Document	In-memory	Graph	Time-series	Ledger
Referential integrity, ACID transactions, schema-on-write	High throughput, low-latency reads and writes, endless scale	Store documents and quickly query on any attribute	Query by key with microsecond latency	Quickly and easily create and navigate relationships between data	Collect, store, and process data sequenced by time	Complete, immutable, and verifiable history of all changes to application data
Lift and shift, ERP, CRM, finance	Real-time bidding, shopping cart, social, product catalog, customer preferences	Content management, personalization, mobile	Leaderboards, real-time analytics, caching	Fraud detection, social networking, recommendation engine	IoT applications, event tracking	Systems of record, supply chain, health care, registrations, financial

AWS: Purpose-built databases



aws SUMMIT

Exploring Each Database

aws SUMMIT

Amazon Relational Database Service (RDS)

Managed relational database service with a choice of six popular database engines

Amazon Aurora · MySQL · PostgreSQL · MariaDB · Microsoft SQL Server · ORACLE

Easy to administer

No need for infrastructure provisioning, installing, and maintaining DB software

Available and durable

Automatic Multi-AZ data replication; automated backup, snapshots, failover

Highly scalable

Scale database compute and storage with a few clicks with no app downtime

Fast and secure

SSD storage and guaranteed provisioned I/O; data encryption at rest and in transit

Traditional SQL

- TCP based wire protocol
- Well Known, lots of uses
- Common drivers (JDBC)
- Frequently used with ORMs
- Scale UP individual instances
- Scale OUT with read replicas
- Sharding at application level
- Lots of flavors but very similar language
- Joins

```
INSERT INTO table1
      (id, first_name, last_name)
VALUES (1, Randall', Hunt');

SELECT col1, col2, col3
FROM table1
WHERE col4 = 1 AND col5 = 2
GROUP BY col1
HAVING count(*) > 1
ORDER BY col2
```

Amazon Aurora

MySQL and PostgreSQL-compatible relational database built for the cloud

Performance and availability of commercial-grade databases at 1/10th the cost

Performance and scalability

5x throughput of standard MySQL and 3x of standard PostgreSQL; scale-out up to 15 read replicas

Availability and durability

Fault-tolerant, self-healing storage; six copies of data across three Availability Zones; continuous backup to Amazon S3

Highly secure

Network isolation, encryption at rest/transit

Fully managed

Managed by RDS: No hardware provisioning, software patching, setup, configuration, or backups

SQL vs NoSQL

SQL	NoSQL
Optimized for storage	**Optimized for compute**
Normalized/relational	Denormalized/hierarchical
Ad hoc queries	Instantiated views
Scale vertically	Scale horizontally
Good for OLAP	Built for OLTP at scale

Amazon DynamoDB

Fast and flexible key value database service for any scale

Performance at scale

Consistent, single-digit millisecond response times at any scale; build applications with virtually unlimited throughput

Serverless

No server provisioning, software patching, or upgrades; scales up or down automatically; continuously backs up your data

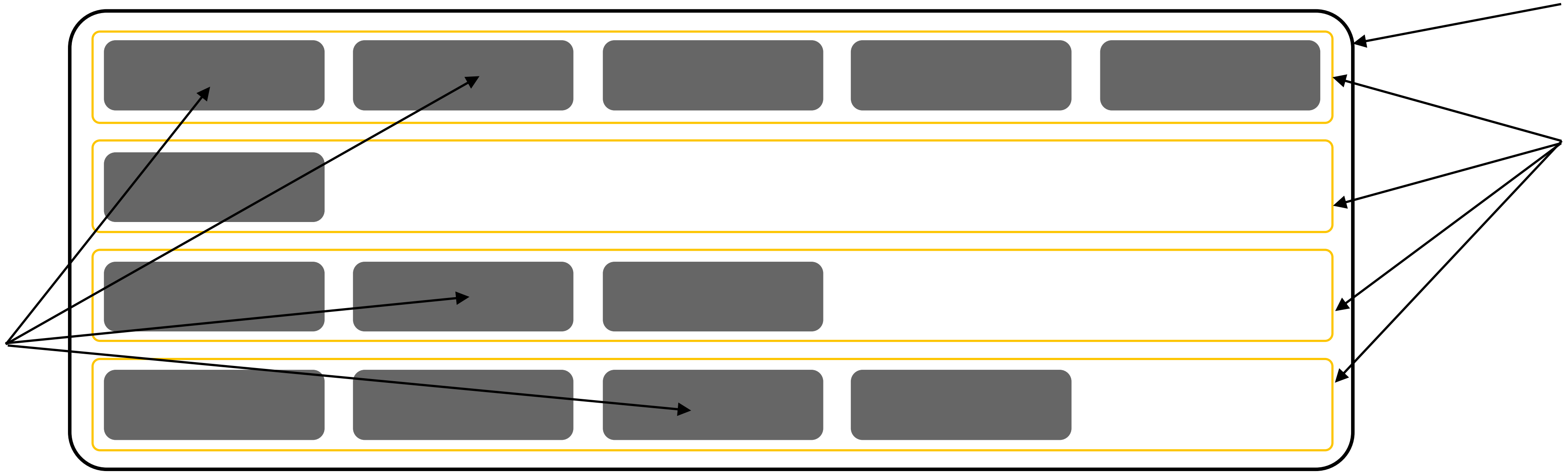
Comprehensive security

Encrypts all data by default and fully integrates with AWS Identity and Access Management for robust security

Global database for global users and apps

Build global applications with fast access to local data by easily replicating tables across multiple AWS Regions

Amazon DynamoDB Data Structure



Partition Key

Sort Key

Mandatory
Key-value access pattern
Determines data distribution

Optional
Model 1:N relationships
Enables rich query capabilities

All items for key
==, <, >, >=, <=
"begins with"
"between"
"contains"
"in"
sorted results
counts
top/bottom N values

aws SUMMIT

DynamoDB Schema and Queries

- Connects over HTTP
- Global Secondary Indexes and Local Secondary Indexes
- Speed up queries with DAX
- Global tables (mutli-region-multi-master)
- Transactions across multiple tables
- Change Streams
- Rich query language with expressions
- Provision read and write capacity units separately with or without autoscaling
- Also supports pay per request model

```
import boto3
votes_table = boto3.resource('dynamodb').Table('votes')
resp = votes_table.update_item(
    Key={'name': editor},
    UpdateExpression="ADD votes :incr",
    ExpressionAttributeValues={":incr": 1},
    ReturnValues="ALL_NEW"
)
```

DynamoDB

Advancements over the last 22 months

February 2017
Time To Live (TTL)

April 2017
VPC endpoints

April 2017
DynamoDB Accelerator (DAX)

June 2017
Auto scaling

November 2017
Global tables

November 2017
On-demand backup

November 2017
Encryption at rest

March 2018
Point-in-time recovery

June 2018
SLA
99.999% SLA

August 2018
Adaptive capacity

November 2018
ACID
Transactions

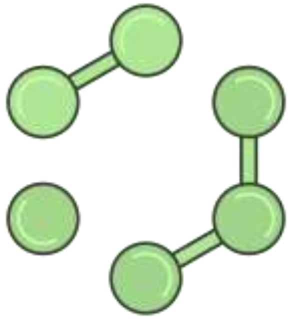
November 2018
On-demand

aws SUMMIT

Amazon DocumentDB: Modern cloud-native architecture

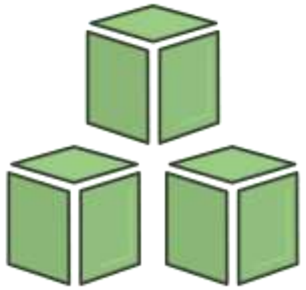
What would you do to improve scalability and availability?

1



Decouple compute and storage

2



Distribute data in smaller partitions

3

Increase the replication of data (6x)

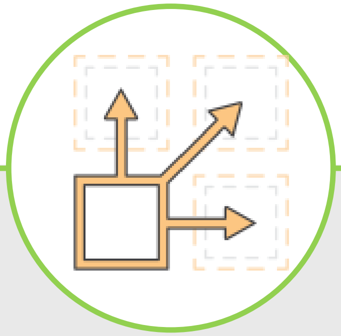
Amazon DocumentDB

Fast, scalable, and fully managed MongoDB-compatible database service

Fast

Millions of requests per second with millisecond latency; twice the throughput of MongoDB

Scalable



Separation of compute and storage enables both layers to scale independently; scale out to 15 read replicas in minutes

Fully managed

Managed by AWS: no hardware provisioning; auto patching, quick setup, secure, and automatic backups

MongoDB compatible

Compatible with MongoDB 3.6; use the same SDKs, tools, and applications with Amazon DocumentDB

Document databases

- Data is stored in JSON-like documents
- Documents map naturally to how humans model data
- Flexible schema and indexing
- Expressive query language built for documents (ad hoc queries and aggregations)

JSON documents are first-class objects of the database

```
{
  id: 1,
  name: "sue",
  age: 26,
  email: "sue@example.com",
  promotions: ["new user", "5%", "dog lover"],
  memberDate: 2018-2-22,
  shoppingCart: [
    {product:"abc", quantity:2, cost:19.99},
    {product:"edf", quantity:3, cost: 2.99}
  ]
}
```

aws SUMMIT

Amazon ElastiCache

Redis and Memcached compatible, in-memory data store and cache

Redis & Memcached compatible

Fully compatible with open source Redis and Memcached

Extreme performance

In-memory data store and cache for microsecond response times

Secure and reliable

Network isolation, encryption at rest/transit, HIPAA, PCI, FedRAMP, multi AZ, and automatic failover

Easily scalable

Scale writes and reads with sharding and replicas

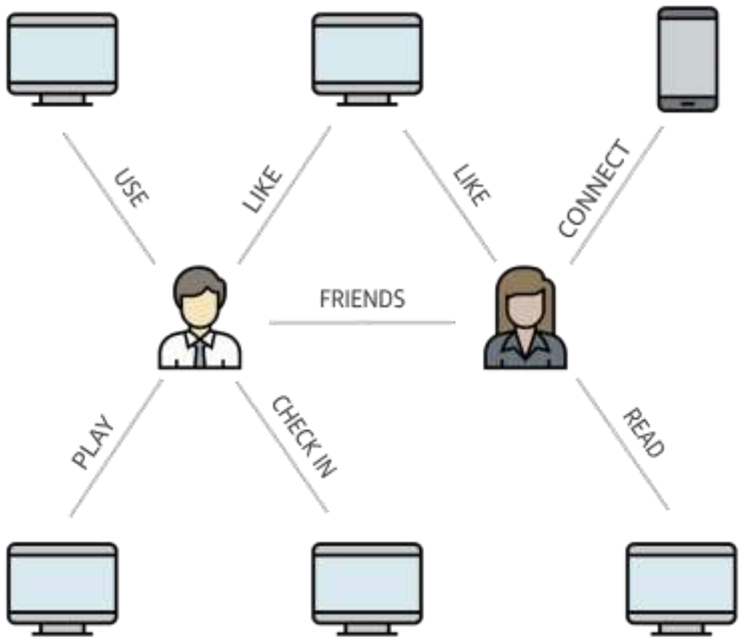
Redis

- Redis Serialization Protocol over TCP (RESP)
- Supports Strings, Hashes, Lists, Sets, and Sorted Sets
- Simple commands for manipulating in memory data structures
- Pub/Sub features
- Supports clustering via partitions

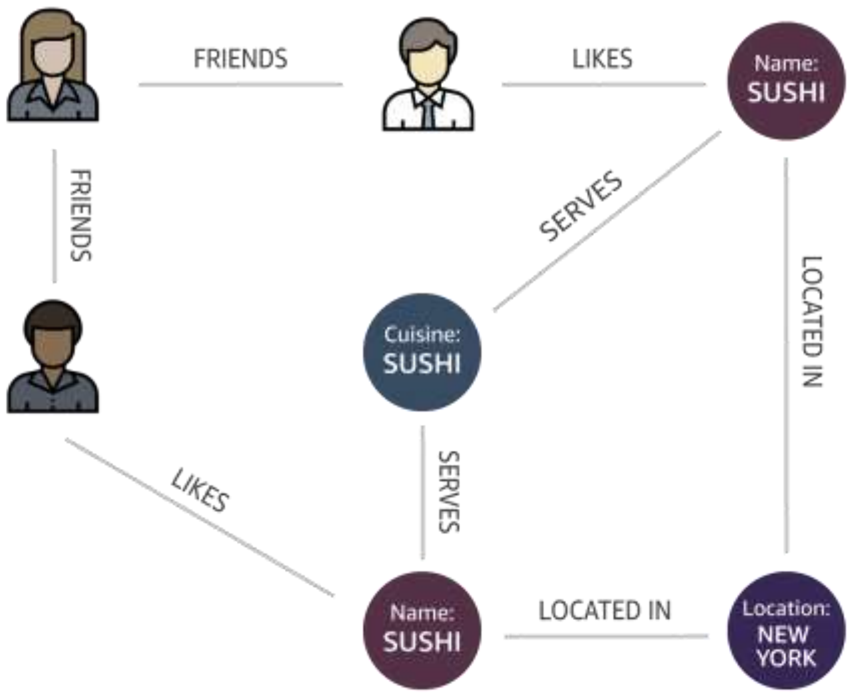
Memcached

- Text and Binary protocols over TCP
- Small number of commands: set, add, replace, append, prepend, cas, get, gets, delete, incr,decr
- Client/Application based partitioning

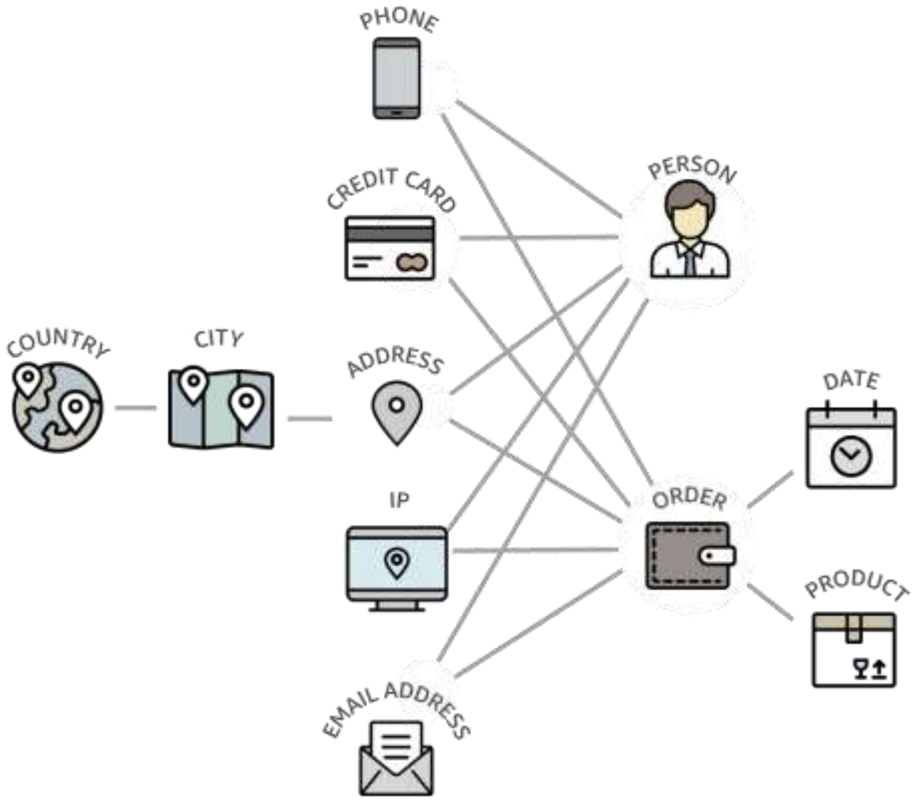
Relationships enable new applications



Social networks



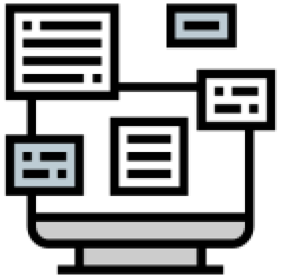
Restaurant recommendations



Retail fraud detection

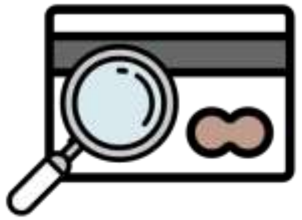
Use cases for highly connected data

Social networking

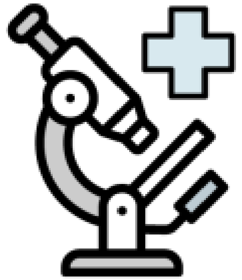


Recommendations

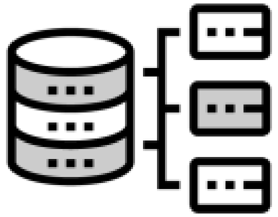
Knowledge graphs



Fraud detection

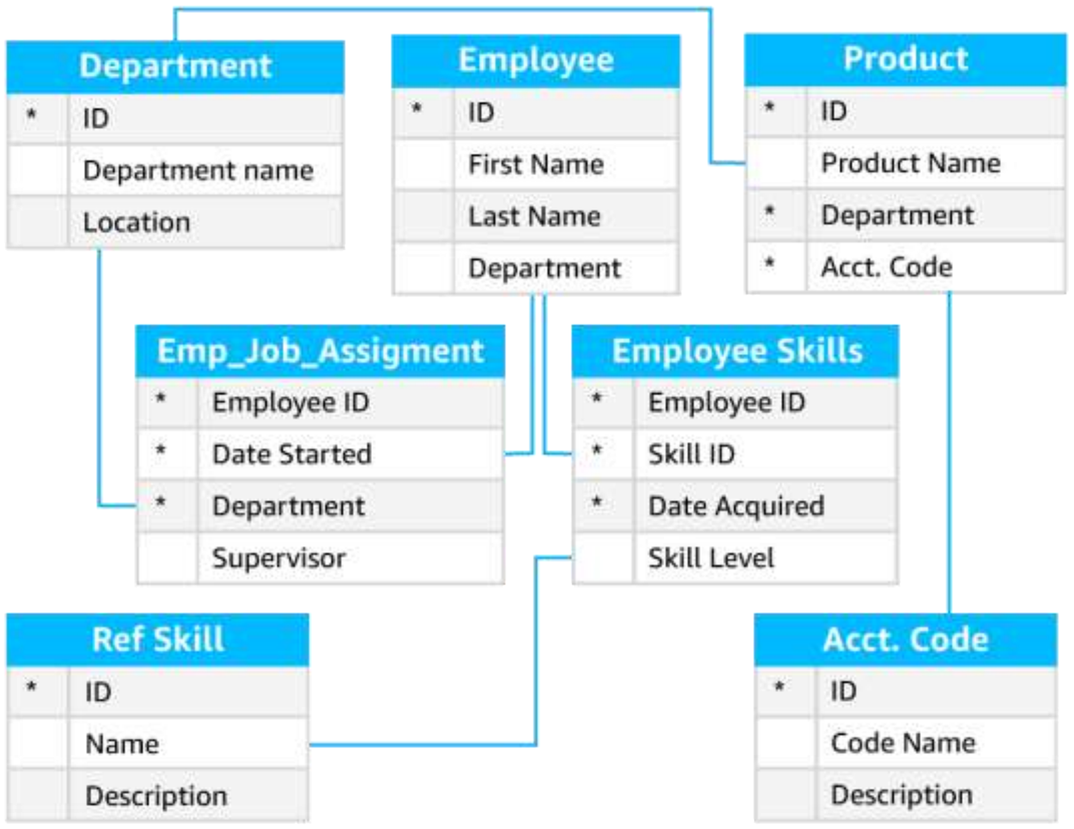


Life Sciences

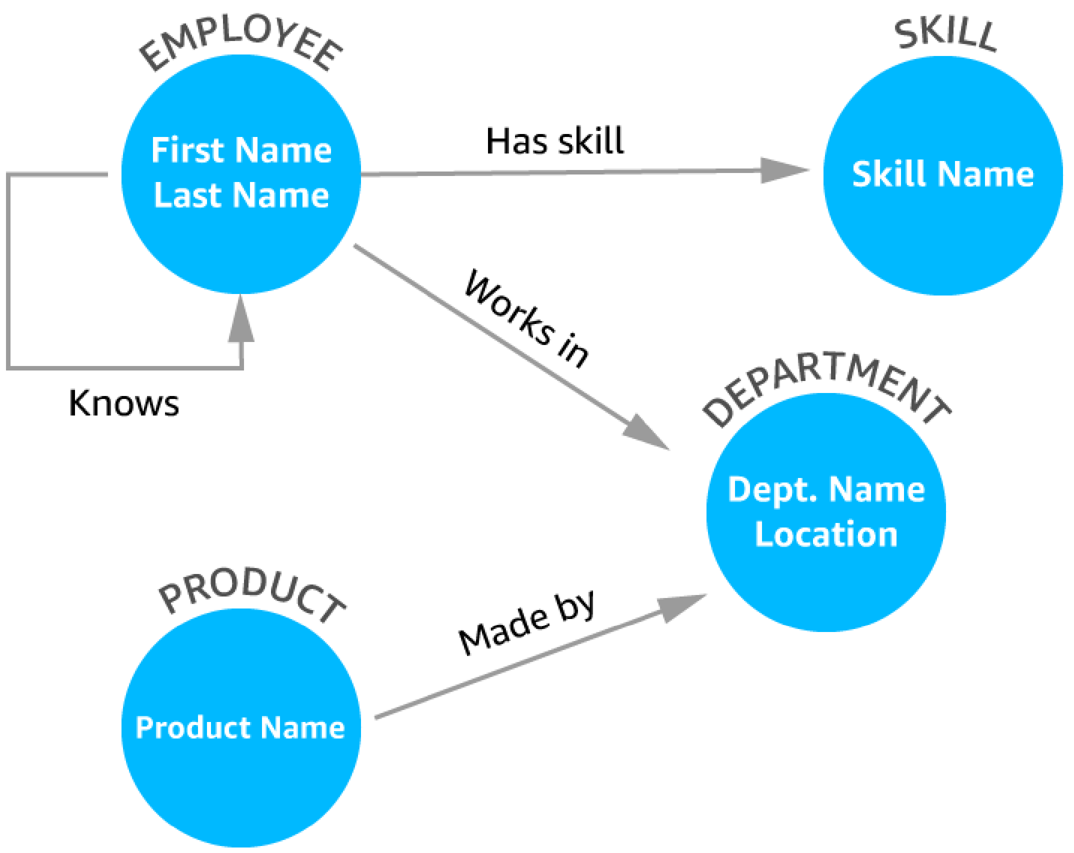


Network & IT operations

Different approaches for highly connected data



Purpose-built for a business process



Purpose-built to answer questions about relationships

Amazon Neptune

Fully managed graph database

Fast

Query billions of relationships with millisecond latency

Reliable

Six replicas of your data across three AZs with full backup and restore

Easy

Build powerful queries easily with Gremlin and SPARQL

Open

Supports Apache TinkerPop & W3C RDF graph models

LEADING GRAPH MODELS AND FRAMEWORKS

PROPERTY GRAPH

Open Source Apache TinkerPop™

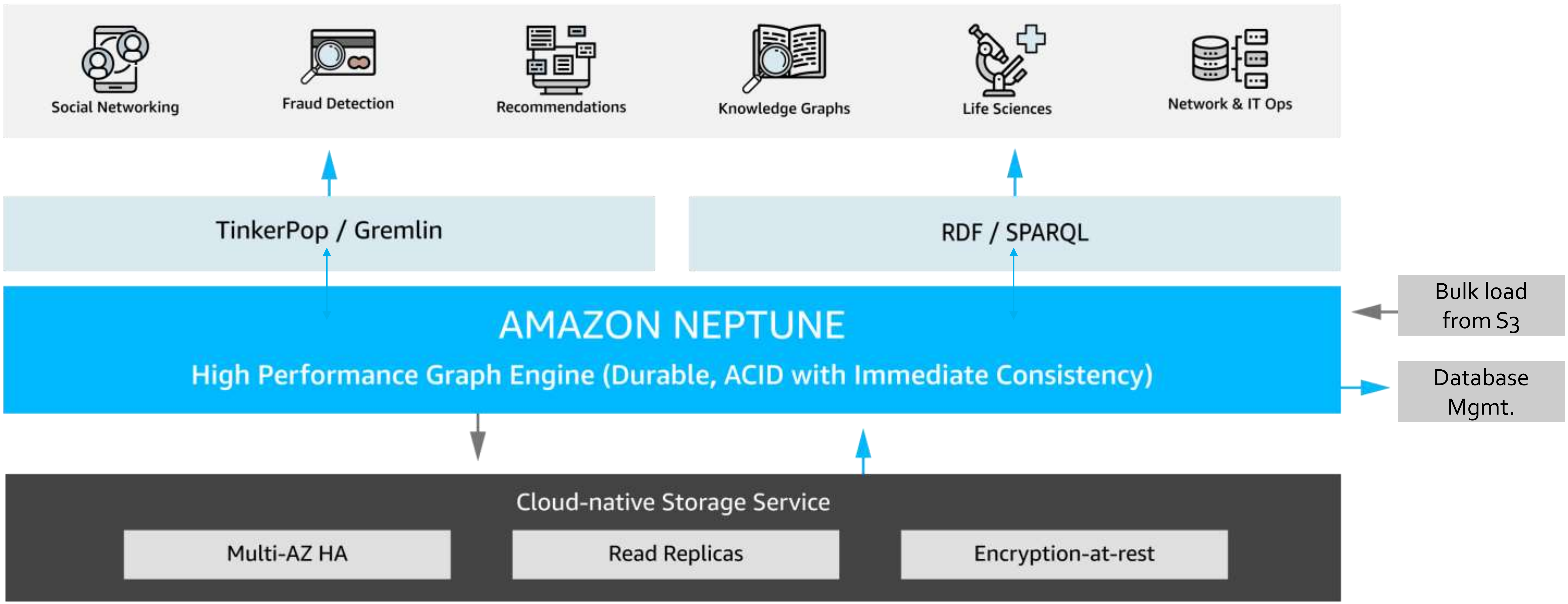
Gremlin Traversal Language

RESOURCE DESCRIPTION FRAMEWORK (RDF)

W3C Standard

SPARQL Query Language

Amazon Neptune high level architecture



Gremlin

```
g.addV('person').property(id, 1).property('name', 'randall')

g.V('1').property(single, 'age', 27)

g.addV('person').property(id, 2).property('name', 'markus')

g.addE('knows').from(g.V('1')).to(g.V('2')).property('weight', 1.0)

g.V().hasLabel('person')

g.V().has('name', 'randall').out('knows').valueMap()
```

http://tinkerpop.apache.org/docs/current/reference/#graph-traversal-steps

SPARQL and RDF

Data:

```
<http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> "SPARQL Tutorial" .
```

Query:

```
SELECT ?title
WHERE
{
  <http://example.org/book/book1> <http://purl.org/dc/elements/1.1/title> ?title .
}
```

This query, on the data above, has one solution:

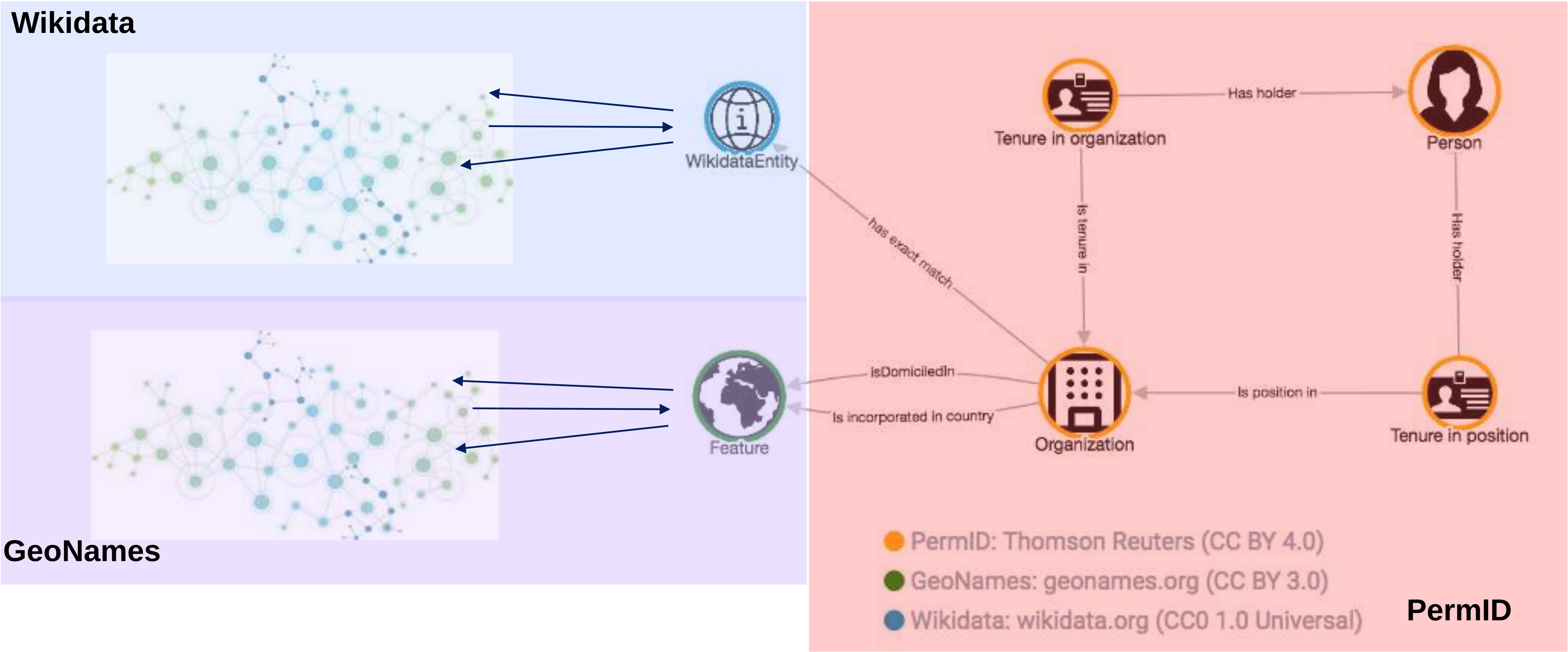
Query Result:

title
"SPARQL Tutorial"

https://www.w3.org/TR/sparql11-query/

THE BENEFIT OF URIs: LINKED DATA

Linking across datasets by referencing globally unique URIs



Example: PermID (re)uses `<http://sws.geonames.org/6252001/>` as a global Identifier for the USA, which is an identifier rooted in GeoNames.

aws SUMMIT

Amazon Timestream (sign up for the preview)

NEW!

Fast, scalable, fully managed time-series database

1,000x faster and 1/10th the cost of relational databases

Collect data at the rate of millions of inserts per second (10M/second)

Trillions of daily events

Adaptive query processing engine maintains steady, predictable performance

Time-series analytics

Built-in functions for interpolation, smoothing, and approximation

Serverless

Automated setup, configuration, server provisioning, software patching

Amazon Quantum Ledger Database (QLDB) (Preview)

NEW!

Fully managed ledger database

Track and verify history of all changes made to your application's data

Immutable

Maintains a sequenced record of all changes to your data, which cannot be deleted or modified; you have the ability to query and analyze the full history

Cryptographically verifiable

Uses cryptography to generate a secure output file of your data's history

Highly scalable

Executes 2–3X as many transactions than ledgers in common blockchain frameworks

Easy to use

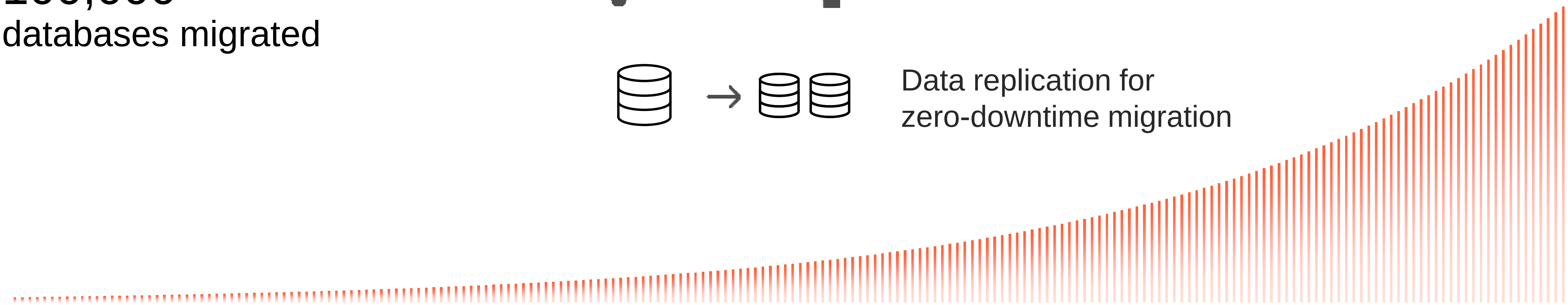
Easy to use, letting you use familiar database capabilities like SQL APIs for querying the data

AWS Database Migration Service (AWS DMS)

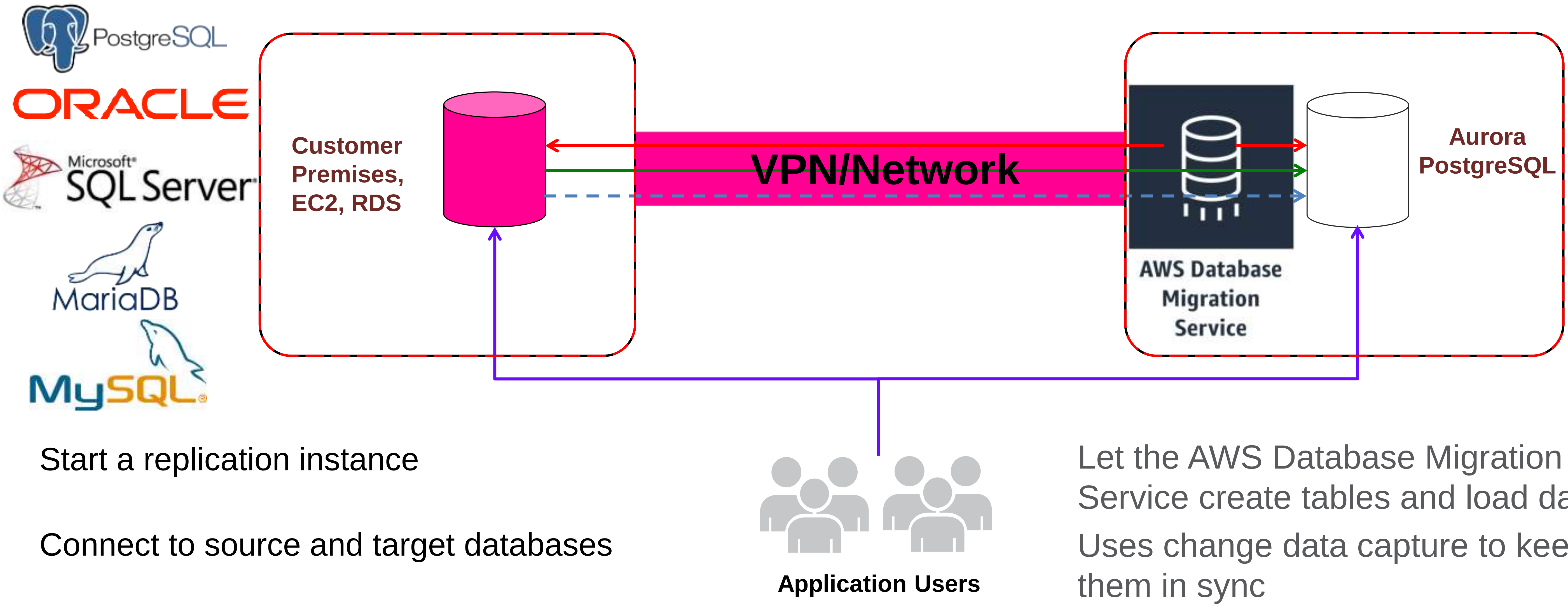
MIGRATING DATABASES TO AWS

100,000+
databases migrated

- Migrate between on-premises and AWS
- Migrate between databases
- Automated schema conversion
- Data replication for zero-downtime migration



AWS DMS—Logical replication



Start a replication instance

Connect to source and target databases

Select tables, schemas, or databases

Let the AWS Database Migration Service create tables and load data

Uses change data capture to keep them in sync

Switch applications over to the target at your convenience

How do we use these in Real Life?

Customers are moving to AWS Databases

Verizon is migrating over 1,000 business-critical applications and database backend systems to AWS, several of which also include the migration of production databases to Amazon Aurora.

By December 2018, Amazon.com will have migrated 88% of their Oracle DBs (and 97% of critical system DBs) moved to Amazon Aurora and Amazon DynamoDB. They also migrated their 50 PB Oracle Data Warehouse to AWS (Amazon S3, Amazon Redshift, and Amazon EMR).

Trimble migrated their Oracle databases to Amazon RDS and project they will pay about 1/4th of what they paid when managing their private infrastructure.

Wappa migrated from their Oracle database to Amazon Aurora and improved their reporting time per user by 75 percent.

Samsung Electronics migrated their Cassandra clusters to Amazon DynamoDB for their Samsung Cloud workload with 70% cost savings.

Intuit migrated from Microsoft SQL Server to Amazon Redshift to reduce data-processing timelines and get insights to decision makers faster and more frequently.

Equinox Fitness migrated its Teradata on-premises data warehouse to Amazon Redshift. They went from static reports to a modern data lake that delivers dynamic reports.

Eventbrite moved from Cloudera to Amazon EMR and were able to cut costs dramatically, spinning clusters up/down on-demand and using Spot (saving > 80%) and Reserved Instances.

Most enterprise database & analytics cloud customers

Most startup database & analytics cloud customers

aws SUMMIT

Markus Ostertag
CEO @ Team Internet AG

aws SUMMIT

Problem: Matching IPs to Ranges in Real-Time Bidding

```
18.215.20.1
52.196.10.4
54.155.40.2
```

IP in range/list?

```
18.208.0.0/13,us-east-1
52.194.0.0/15,ap-northeast-1
54.155.0.0/16,eu-west-1
```

Real-Time Bidding Requirements:

- High throughput (>25000 req/s)
- Response time is critical
- Scalability

Possible choices

	Relational	Key-value	In-memory
Upside	Well known, Flexible Indices, Query language, Data scheme obvious	High throughput, low-latency reads, endless scale	Query by key with microsecond latency
Downside	Latency? Query volume? Future scale?	Data scheme? Indices? Sharding?	Data scheme? Sharding? Future scale?

Possible choices

	Relational	Key-value	In-memory
Upside	Well known, Flexible Indices, Query language, Data scheme obvious	High throughput, low-latency reads, endless scale	Query by key with microsecond latency
Downside	Latency? Query volume? Future scale?	Data scheme? Indices? Sharding?	Data scheme? Sharding? Future scale?

DynamoDB?

But how?

- Sharding and Index downsides addressed by transforming data during ingest
- Accept multiple entries per range -> sharding & query possible/optimized

```
18.208.0.0/13,us-east-1
52.194.0.0/15,ap-northeast-1
54.155.0.0/16,eu-west-1
```

Partition key | Sort key

ip_net	ip_end	ip_start	region
18.208	315686911	315621376	us-east-1
18.209	315752447	315686912	us-east-1
52.194	885194751	885129216	ap-northeast-1
52.195	885260287	885194752	ap-northeast-1
54.155	916193279	916127744	eu-west-1

DynamoDB it is!

```
var _findIp = function(ip, ipLong, dynamo, callback) {
  var ip_parts = ip.split(".");
  var net_string = ip_parts[0]+"."+ip_parts[1];
  var params = {
    TableName: "ip-ranges",
    KeyConditionExpression:
      "ip_net = :net and ip_end >= :ipLong",
    FilterExpression: "ip_start <= :ipLong",
    ExpressionAttributeValues: {
      ":net": {S: net_string},
      ":ipLong": {N: ipLong.toString()}
    }
  };
  dynamo.query(params, callback);
};
```

ip_net	ip_end	ip_start	region
18.208	315686911	315621376	us-east-1
18.209	315752447	315686912	us-east-1
52.194	885194751	885129216	ap-northeast-1
52.195	885260287	885194752	ap-northeast-1
54.155	916193279	916127744	eu-west-1

DynamoDB it is!

```
var _findIp = function(ip, ipLong, dynamo, callback) {
  var ip_parts = ip.split(".");
  var net_string = ip_parts[0]+"."+ip_parts[1];
  var params = {
    TableName: "ip-ranges",
    KeyConditionExpression:
      "ip_net = :net and ip_end >= :ipLong",
    FilterExpression: "ip_start <= :ipLong",
    ExpressionAttributeValues: {
      ":net": {S: net_string},
      ":ipLong": {N: ipLong.toString()}
    }
  };
  dynamo.query(params, callback);
};
```

ip_net	ip_end	ip_start	region
18.208	315686911	315621376	us-east-1
18.209	315752447	315686912	us-east-1
52.194	885194751	885129216	ap-northeast-1
52.195	885260287	885194752	ap-northeast-1
54.155	916193279	916127744	eu-west-1

DynamoDB it is!

```
var _findIp = function(ip, ipLong, dynamo, callback) {
  var ip_parts = ip.split(".");
  var net_string = ip_parts[0]+"."+ip_parts[1];
  var params = {
    TableName: "ip-ranges",
    KeyConditionExpression:
      "ip_net = :net and ip_end >= :ipLong",
    FilterExpression: "ip_start <= :ipLong",
    ExpressionAttributeValues: {
      ":net": {S: net_string},
      ":ipLong": {N: ipLong.toString()}
    }
  };
  dynamo.query(params, callback);
};
```

ip_net	ip_end	ip_start	region
18.208	315686911	315621376	us-east-1
18.209	315752447	315686912	us-east-1
52.194	885194751	885129216	ap-northeast-1
52.195	885260287	885194752	ap-northeast-1
54.155	916193279	916127744	eu-west-1

aws SUMMIT

Learnings

- Focus on “core requirements“
- Be creative about data schemes
- Accept minor downsides – very often there is no perfect solution
- Use upsides of chosen database to optimize (i.e. adding DAX for caching)

Knowledge Graphs @ Siemens

Sebastian Brandt
Steffen Lamparter

Semantics and Reasoning Group
Siemens Corporate Technology

CT RDA BAM SMR-DE

February 2019

Knowledge graphs become especially powerful for managing complex queries and heterogeneous data

SIEMENS
Ingenuity for life

Why Knowledge Graphs?

- **Graphs are a natural way** to represent entities and their relationships
- Graphs can **capture a broad spectrum of data** (structured / unstructured)
- Graphs can be managed **efficiently**

Game-changing data integration

Robust data quality assurance

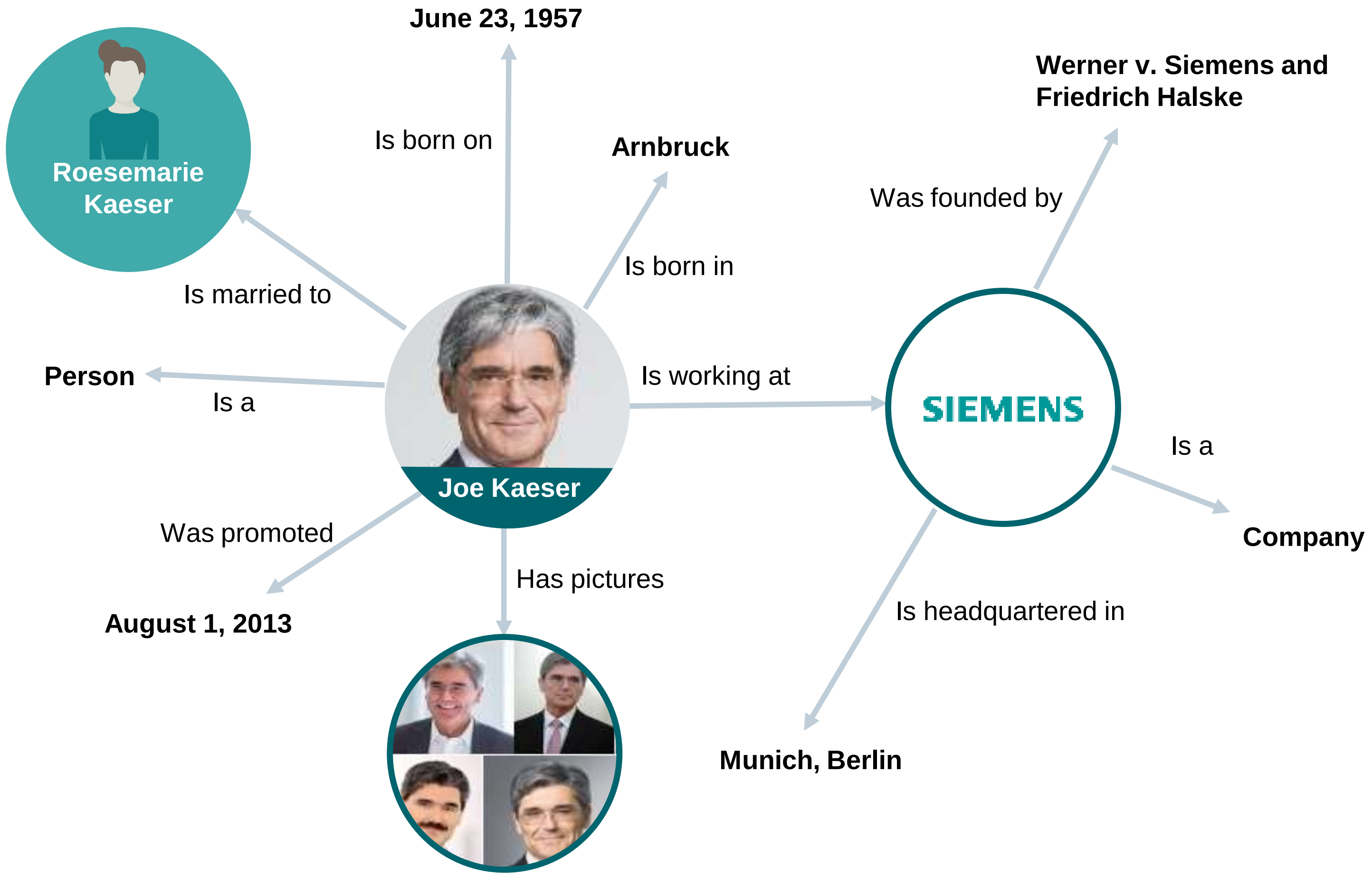
Intuitive domain modelling

Flexibility & performance

Low up-front investment

What are Graphs? Knowledge representation formalism semantic descriptions of entities and their relationships

SIEMENS
Ingenuity for life



Rules make it possible to add further expert knowledge, e.g. "Siemens has to be a company, as a person is working there"

Objects
Real-world objects (things, places, people) and abstract concepts (genres, religions, professions)

Relationships
Logical connection between two objects e.g. Joe Kaeser <u>is born in</u> Arnbruck

Semantic descriptions
The semantic description indicates the meaning of an object or relation, e.g. Joe Kaeser <u>is a person</u>

Knowledge graphs become a powerful addition to traditional data warehouses for managing heterogeneous data with complex relations

SIEMENS
Ingenuity for life

	Traditional data warehouses		Knowledge graphs
Dimensions of data relations	Limited relations across data, e.g. time series data	low ↔ high	High number & complex relations, e.g. social networks
Variety of questions to be answered	"We have a clear scope of user questions" → fixed set of dashboard needed	low ↔ high	"We do not yet know all the different questions, users will ask" → Chatbot
Data heterogeneity	Only few different data types and data sources e.g. invoicing database in ERP	low ↔ high	Analysis needs to combine heterogeneous data, e.g. sensor data, text data, product data
Quality of data	Ensured consistency of stored data	low ↔ high	Need to handle imperfect, incomplete or inconsistent data

Use cases for knowledge graphs can be clustered into five categories – overview and use case examples

SIEMENS
Ingenuity for life

Degree of complexity

Data quality	Data access & dashboarding	Digital companion	Recommender system	Constraints & planning
Improving data availability and quality by combining and comparing data from various sources to fill in missing data sets or identify potentially wrong data and data duplicates	Maintaining up-to-date meta-data, creating transparency on all available data and making them accessible to users via queries	Enhancing features of existing products or services with digital companions that are able to understand and process user questions and providing the needed data insights	Providing users high quality recommendations by identifying similarities in historical data	Enabling autonomous systems to understand data and its dependencies and take own decisions, such as autonomous planning of production proces-ses

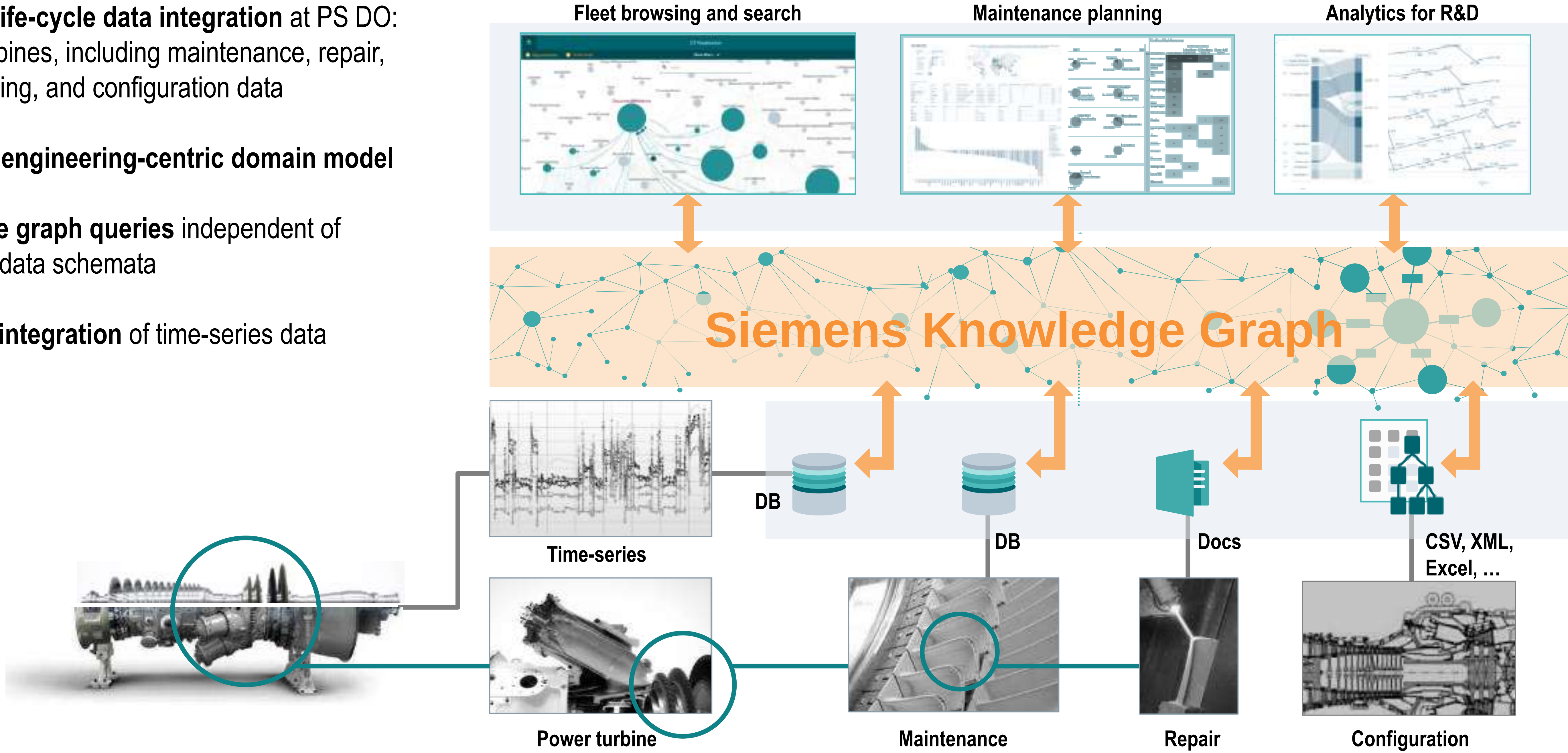




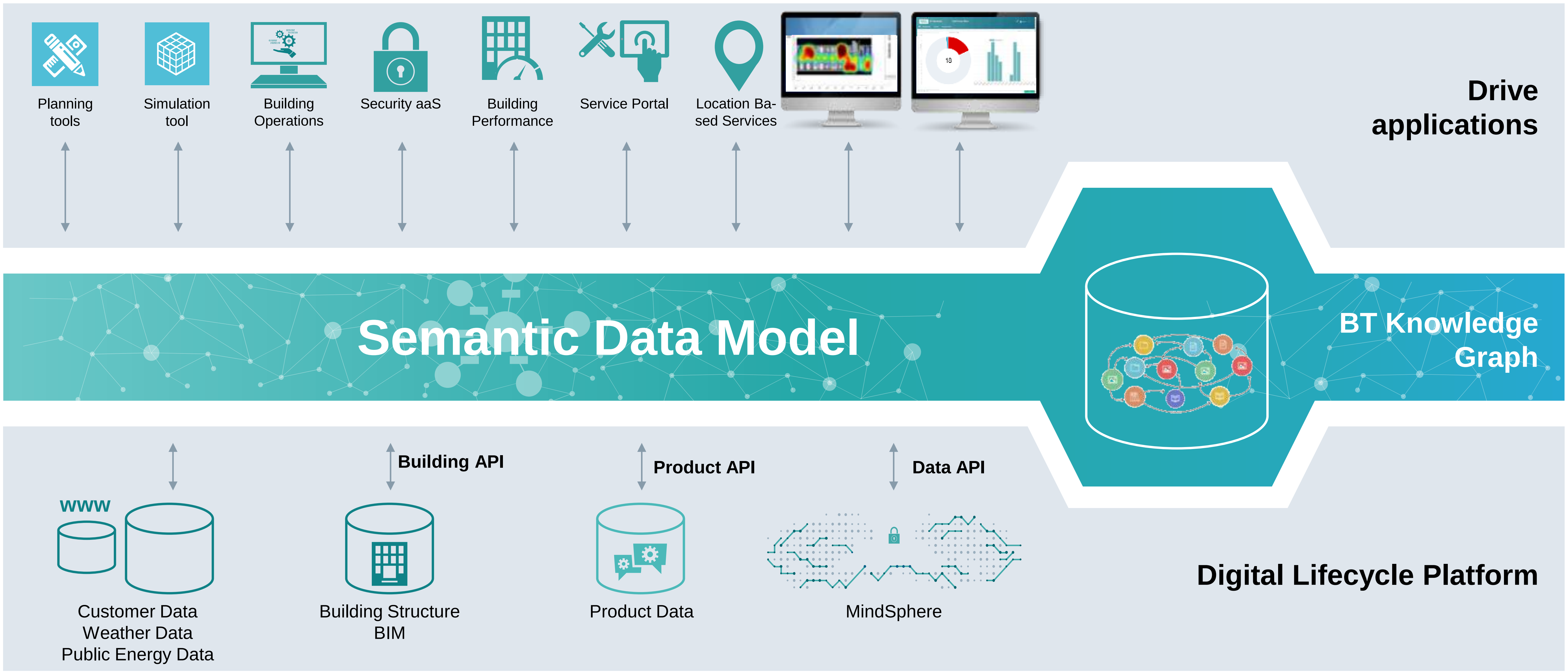
Example: gas-turbine maintenance planning

SIEMENS
Ingenuity for life

- **Cross-life-cycle data integration** at PS DO: gas-turbines, including maintenance, repair, monitoring, and configuration data
- Holistic **engineering-centric domain model**
- **Intuitive graph queries** independent of source data schemata
- **Virtual integration** of time-series data



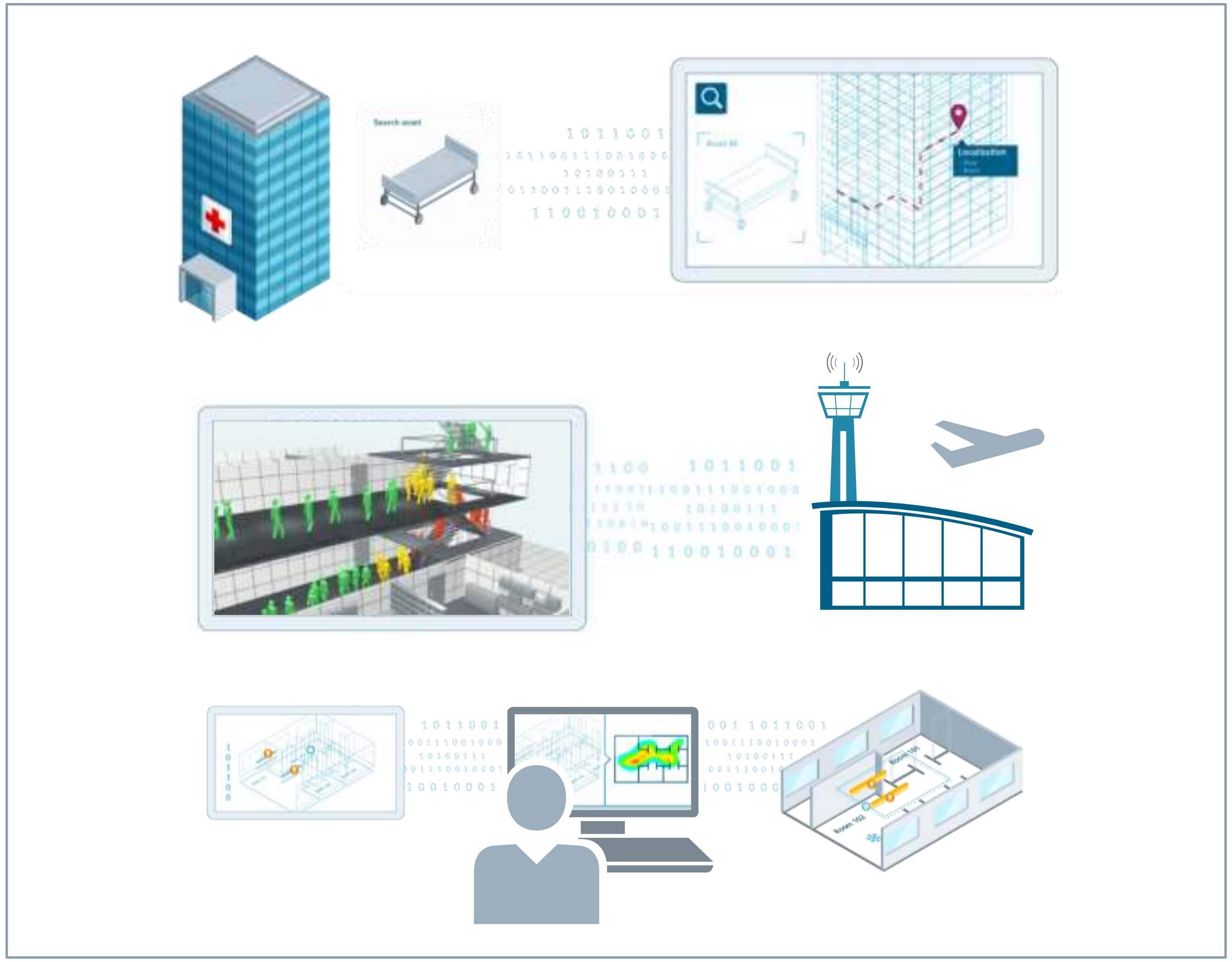
A semantic data model enables flexible linking and an integrated, intuitive API for applications



Creating perfect places based on Services – a user-centric holistic approach to the modern workplace …

SIEMENS
Ingenuity for life

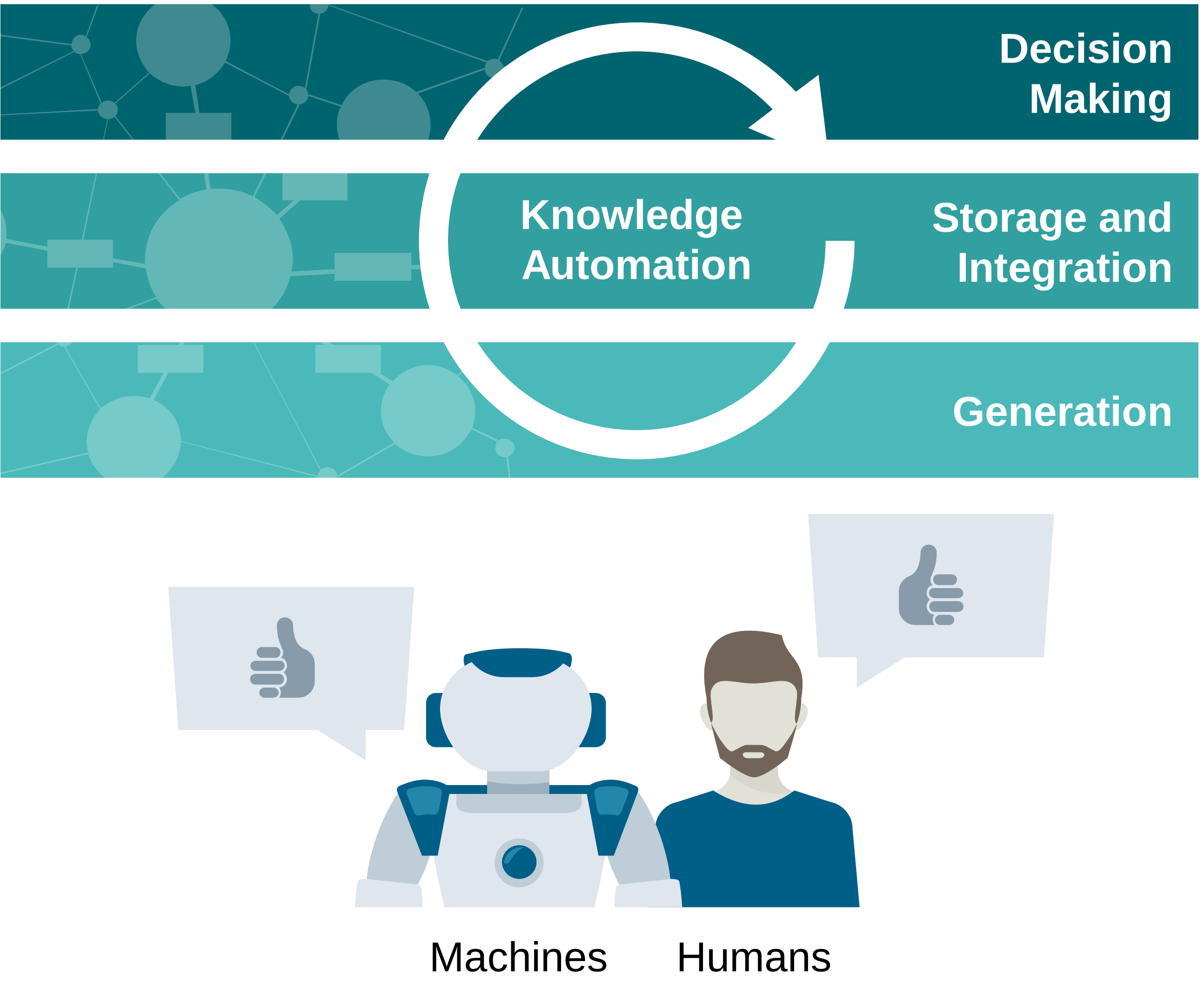
Customer Interest	Relevant KPI's
Energy and asset efficiency	Optimizing CAPEX and OPEX CO_2 emissions Asset Performance/Useful Life
Space efficiency	Cost per space unit Workplace Utilization Revenue per space unit Vacancy Rate
Individual efficiency and comfort	Employee productivity Employee satisfaction



Industrializing Knowledge Graphs

SIEMENS
Ingenuity for life

Industrial Knowledge Graph



Relevant technologies

Decision Making

- Reasoning and Constraint Solving
- Machine/Deep Learning
- Question Answering

Storage and Integration

- Graph/NoSQL databases
- Constraints and Rules
- Probabilistic programming
- Ontologies

Generation

- NLP/Text understanding
- Machine/Deep Learning
- Computer vision
- Sound recognition
- Virtual data Integration
- Information retrieval
- …

R&D Areas

Decision Making

- Explanation of AI decisions
- Data access: Semantic Search
- Machine Learning on Graphs for recommendations, quality, etc.

ML for Graphs

Storage and Integration

- Reusable Semantic Modelling and Knowledge Graphs
- Data integration and cleaning (entity reconciliation)

Ontology Library

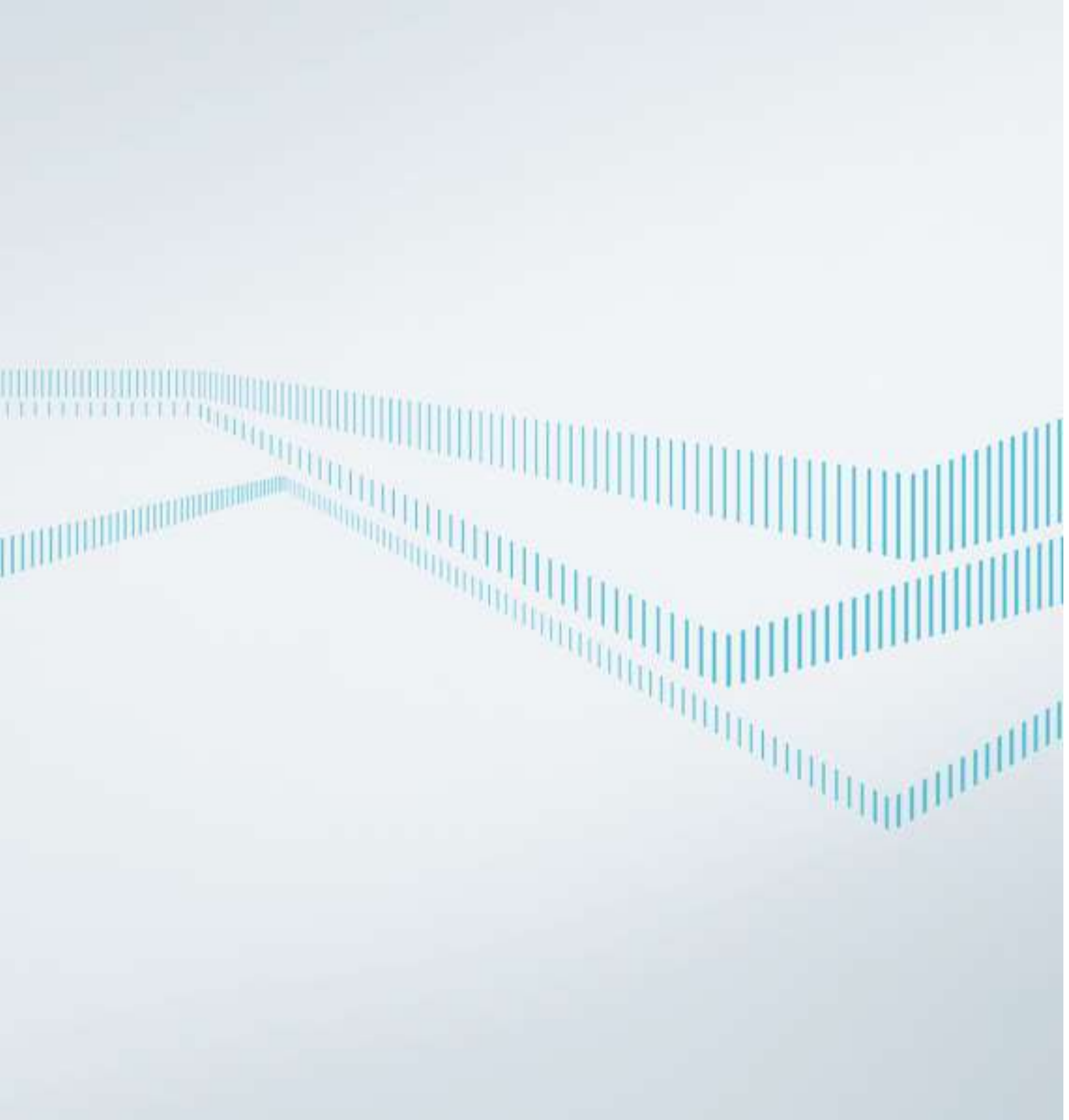
Simple RDF + PGM

Generation

- Extraction from unstructured data (inclusive text, audio, image)
- Automatic semantic annotation of structured data
- Learning of domain-specific rules/patterns

ML for automated annotation

Get in touch!



Dr. Sebastian Brandt

Senior Key Expert – Knowledge Graph and Data Management
CT RDA BAM SMR-DE

Dr. Steffen Lamparter

Head of Research Group Semantics & Reasoning
CT RDA BAM SMR-DE

Siemens AG
Corporate Technology
Otto-Hahn-Ring 6
81739 München Germany

E-mail
steffen.lamparter@siemens.com

Intranet
intranet.siemens.com/ct

Thank you!

aws SUMMIT

Please complete the session survey.

aws SUMMIT