AWS re:Invent

ADM301

Best practices for identity resolution with Amazon Neptune

Nitin Poddar
Lead Software Engineer
Zeta Global

Taylor Riggan
Senior Specialist Solutions Architect
Amazon Neptune
Amazon Web Services

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Related breakouts

- DAT361: Deep dive on Amazon Neptune
  - (Tuesday, 5:30 PM)
- ADM203: Reimagining advertising analytics & identity resolution at scale
  - (Wednesday, 1:00 PM)
- DAT220: Real-world customer use cases with Amazon Neptune
  - (Thursday, 1:45 PM)
- DAT347: Neptune best practices: How to optimize your graph queries
  - DAT347-R (Monday, 5:30 PM)
  - DAT347-R1 (Tuesday, 11:30 AM)
  - DAT347-R2 (Wednesday, 10:00 AM)
  - DAT347-R3 (Thursday, 3:15 PM)

Common data categories and use cases



Relational

Referential integrity, ACID transactions, schema-on-write

Lift and shift, ERP, CRM, finance



Key-value

High-throughput, low-latency reads and writes, highly scalable

Real-time bidding, shopping cart, social, product catalog, customer preferences



Document

Store documents and quickly access querying on any attribute

Content management, personalization, mobile



In-memory

Query by key with microsecond latency

Leaderboards, real-time analytics, caching



Graph

Quickly and easily create and navigate relationships between data

Fraud detection, social networking, recommendation engine



Time-series

Collect, store, and process data sequenced by time

IoT applications, event tracking



Ledger

Complete, immutable, and verifiable history of all changes to application data

Systems of record, supply chain, healthcare, registrations, financial

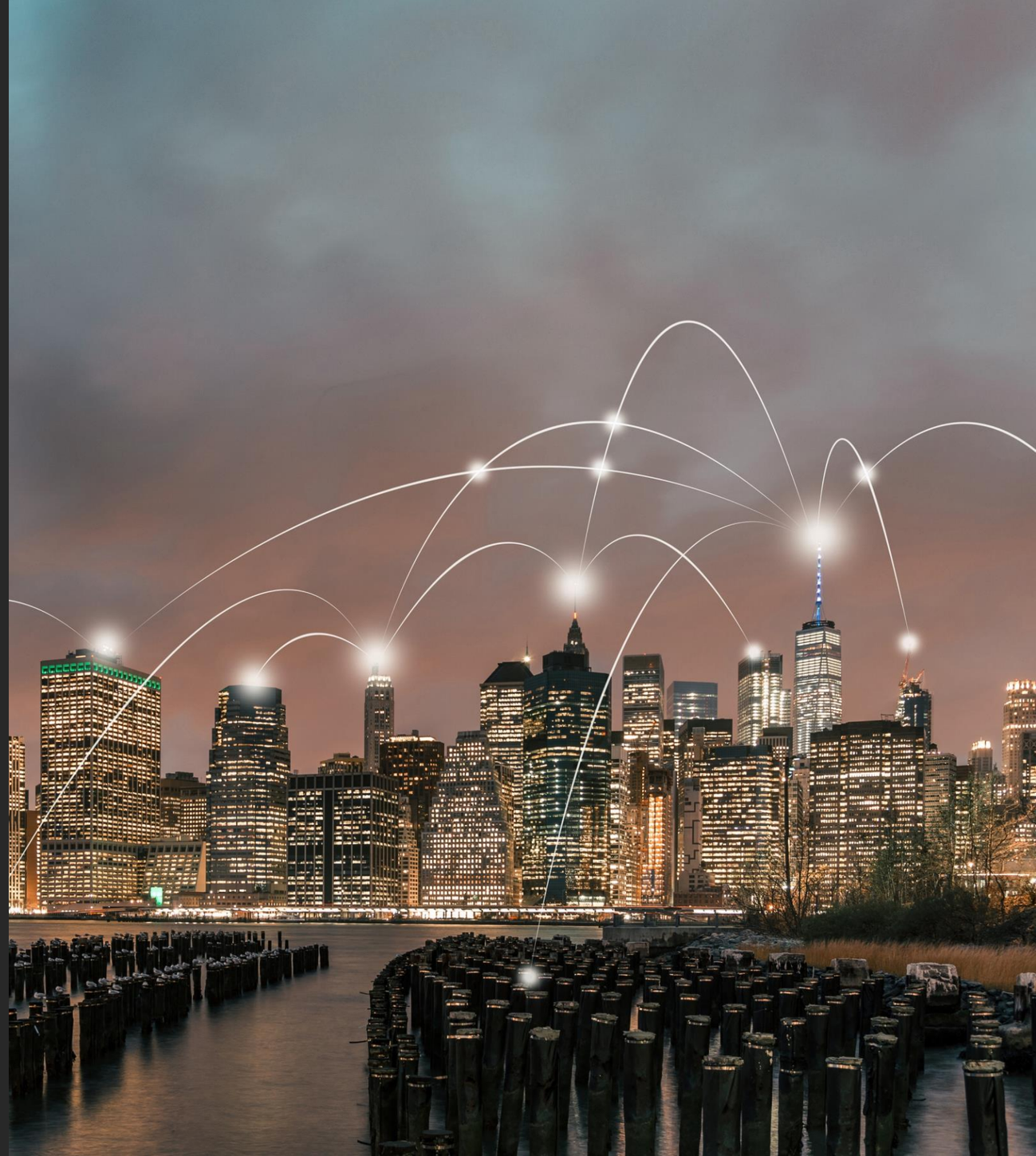
Agenda

- Identity resolution
- IDR architecture
- Why a graph database?
- IDR data model
- Why Amazon Neptune?
- Key consideration & lessons learned

What is identity resolution?

- **Identity** – Anonymous profile of a customer

- **Resolution** – Connecting all the identifiers into a single profile



Why IDR?

01 Personalization

02 Customer experience

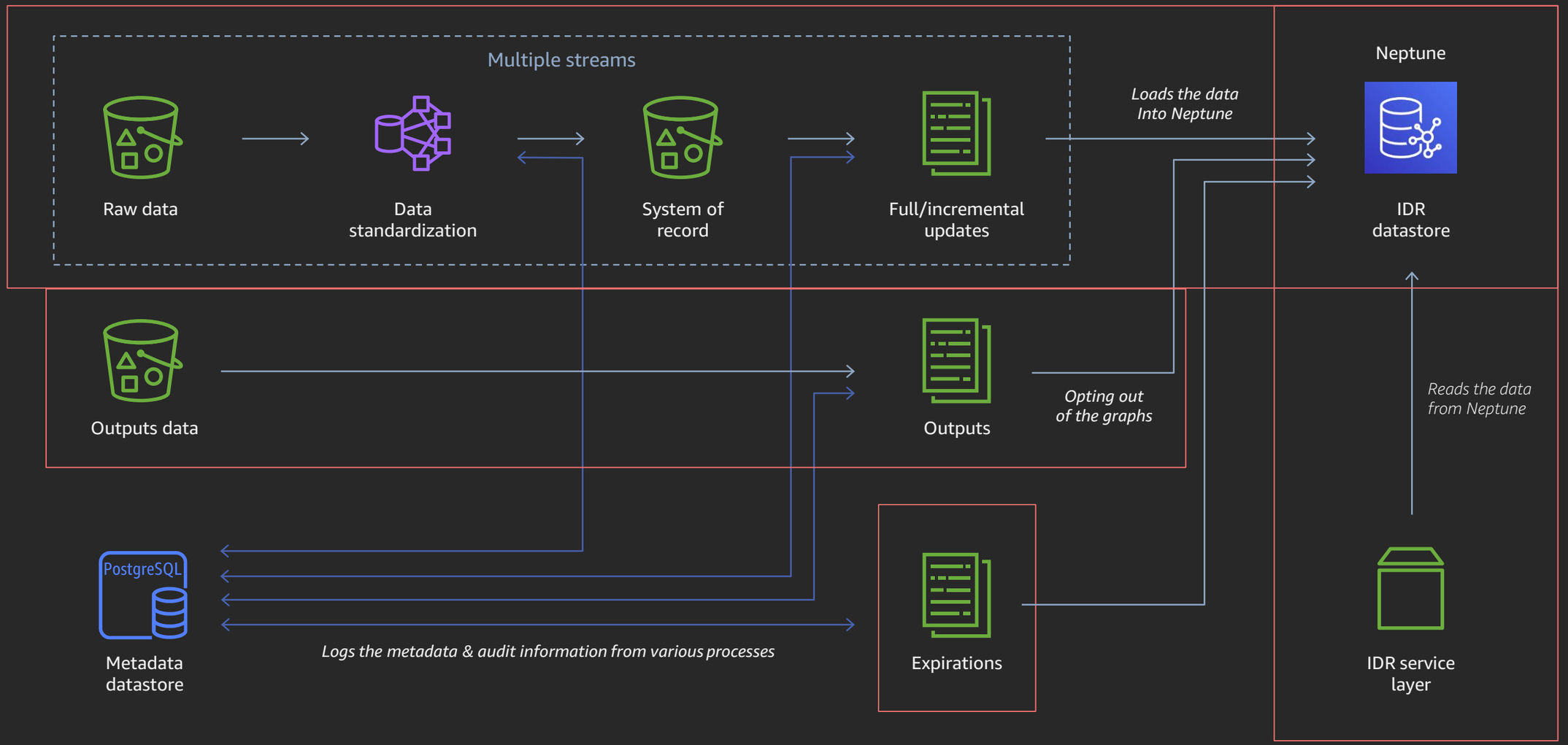
03 Single view of customer

04 Efficient spend

IDR architecture

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

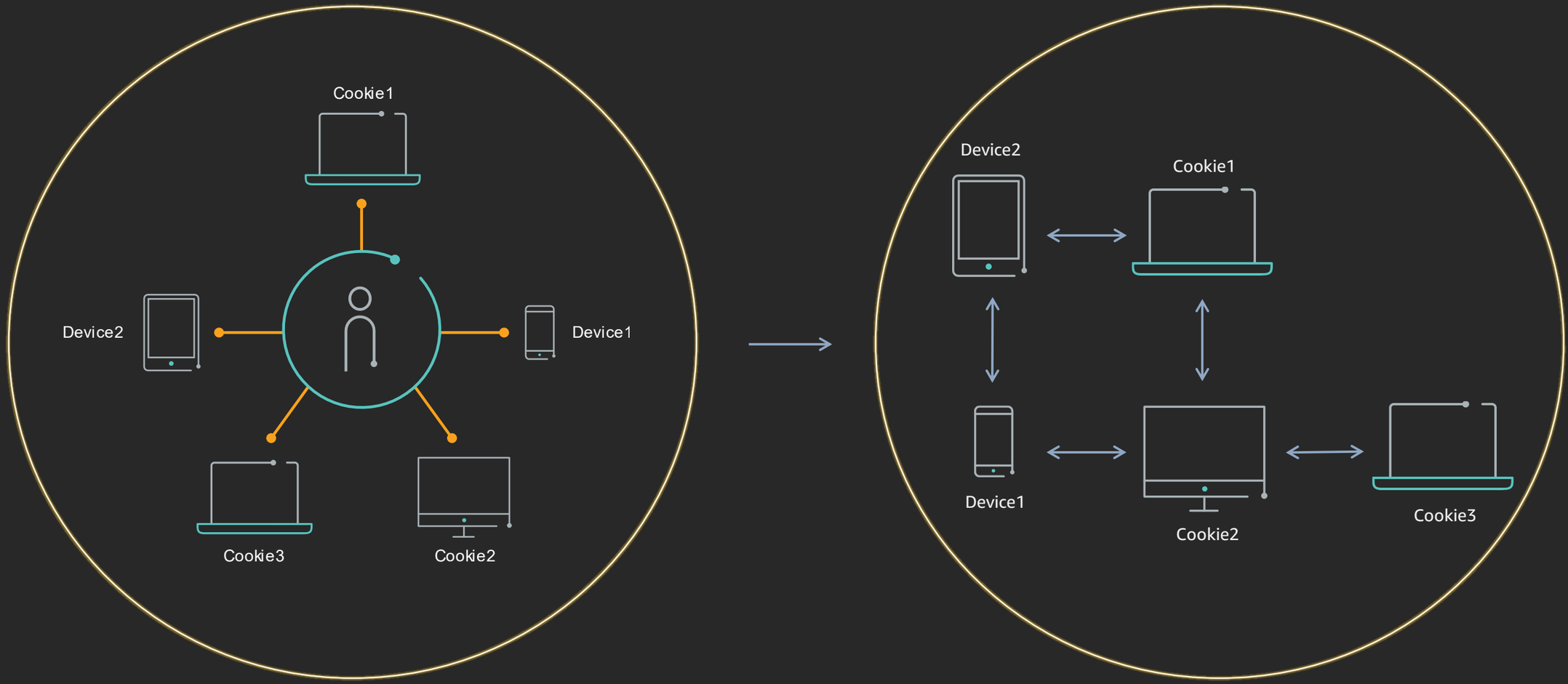
IDR data flow



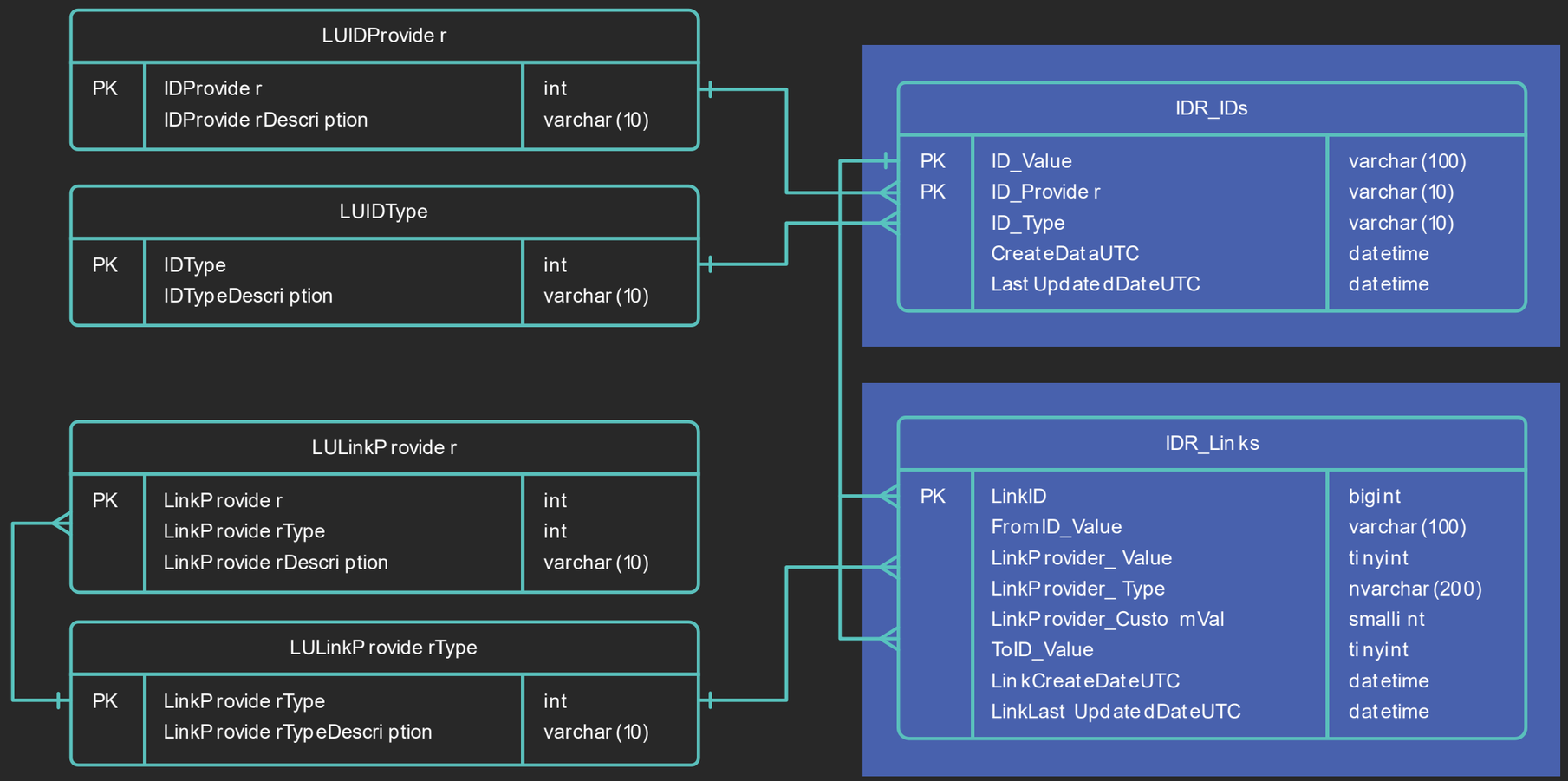
Why a graph database?

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

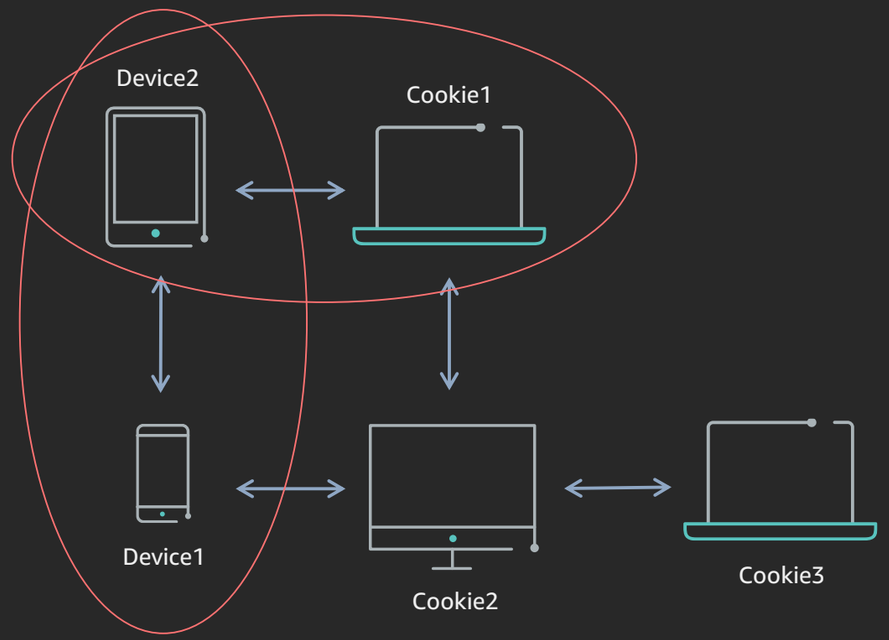
Customer identifiers



SQL model


LUIDProvider
PK
IDProvider
IDProvider Description
int
varchar(10)
LUIDType
PK
IDType
IDTypeDescription
int
varchar(10)
LULinkProvider
PK
LinkProvider
LinkProviderType
LinkProviderDescription
int
int
varchar(10)
LULinkProviderType
PK
LinkProviderType
LinkProviderTypeDescription
int
varchar(10)
IDR_IDs
PK
PK
ID_Value
ID_Provider
ID_Type
CreateDataUTC
LastUpdatedDateUTC
varchar(100)
varchar(10)
varchar(10)
datetime
datetime
IDR_Links
PK
LinkID
FromID_Value
LinkProvider_Value
LinkProvider_Type
LinkProvider_CustomVal
ToID_Value
LinkCreateDateUTC
LinkLastUpdatedDateUTC
bigint
varchar(100)
tinyint
nvarchar(200)
smallint
tinyint
datetime
datetime

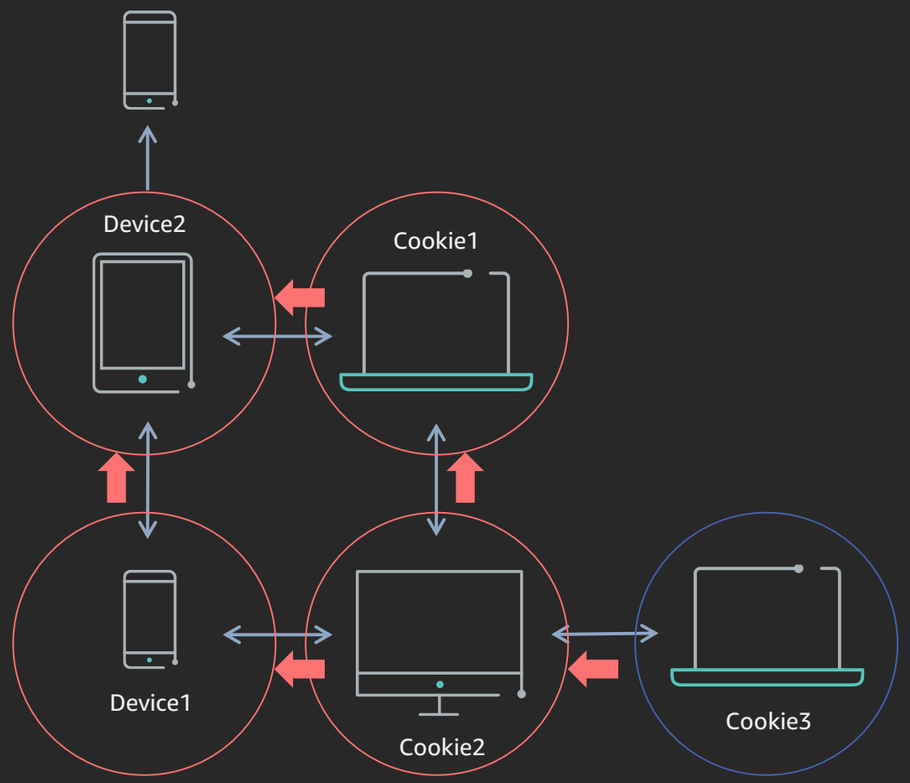
Sample data model



ID	ID provider	ID type
Cookie1	iOne	Cookie
Cookie2	iOne	Cookie
Cookie3	iOne	Cookie
Device1	AAID	Mobile
Device2	IDFA	Mobile

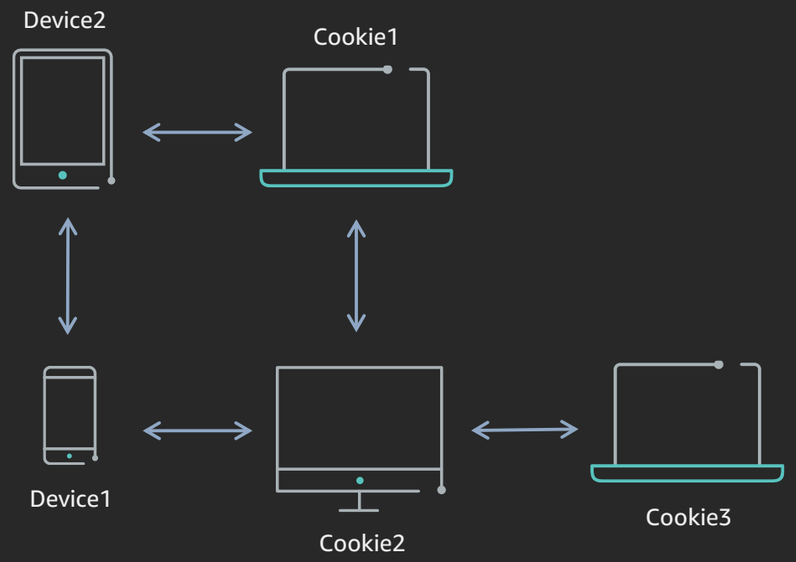
Form ID	To ID	Link provider type
Device2	Cookie1	Deterministic
Cookie1	Device2	Deterministic
Device2	Device1	Deterministic
Device1	Device2	Deterministic
Device1	Cookie3	Deterministic
Cookie3	Device1	Deterministic
Cookie3	Cookie2	Probabilistic
Cookie2	Cookie3	Probabilistic
Cookie1	Cookie3	Probabilistic
Cookie3	Cookie1	Probabilistic

SQL query



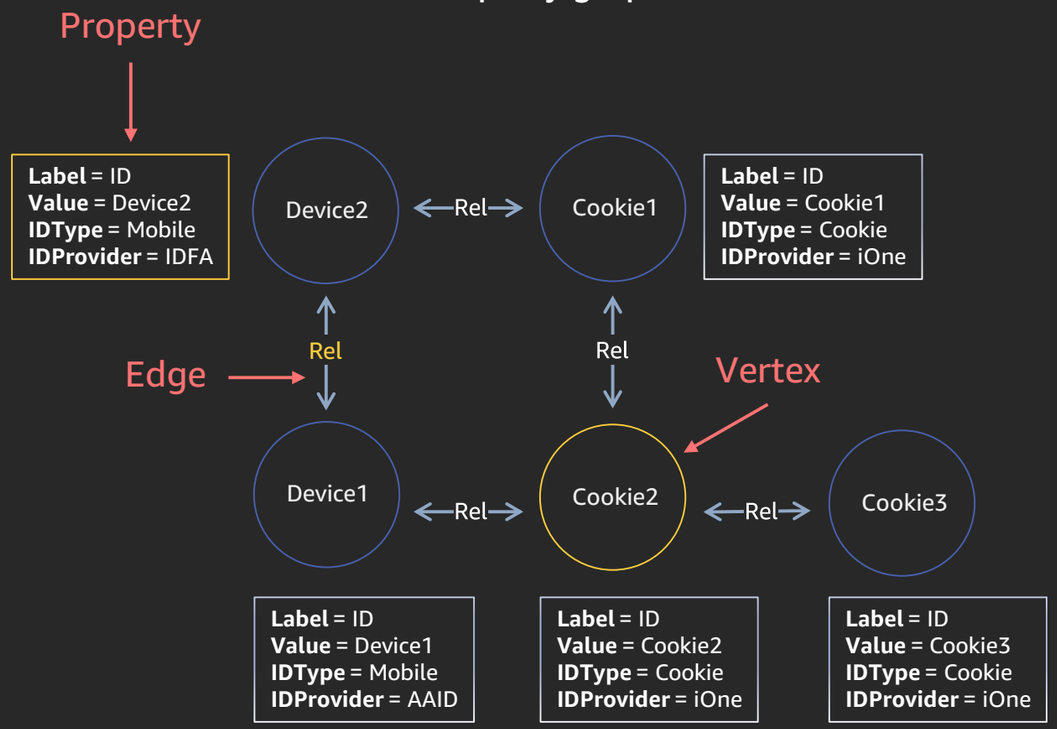

```
With CTE_Level1 as
(
          SELECT ToID
          FROM
          IDR_Links WHERE FromID ='Cookie3'
),
 CTE_Level2 as
(
          SELECT ToID
          FROM
          IDR_Links WHERE FromID IN ( SELECT ToID FROM CTE_Tier1 )
),
CTE_Level3 as
(
          SELECT ToID
          FROM
          IDR_Links WHERE FromID IN( SELECT ToID FROM CTE_Tier2 )
)
SELECT ToID FROM CTE_Level1
Union
SELECT ToID FROM CTE_Level2
Union
SELECT ToID FROM CTE_Level3
```

Customer identifiers



Graph model

Property graph model



Graph vs. RDBMS

Easy to model for highly connected data	⟷	Complex models
Relationships are first-class citizens	⟷	Relationships are represented by foreign keys
Imperative Gremlin	⟷	Declarative SQL
Graphs are flexible	⟷	Rigid SQL schema
Graph traversal performance	⟷	SQL query performance

Why Amazon Neptune?

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Neptune



No hardware management



Instant provisioning



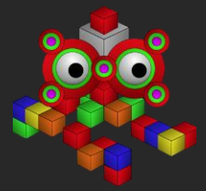
Scaling



Cost effective



Security & compliance

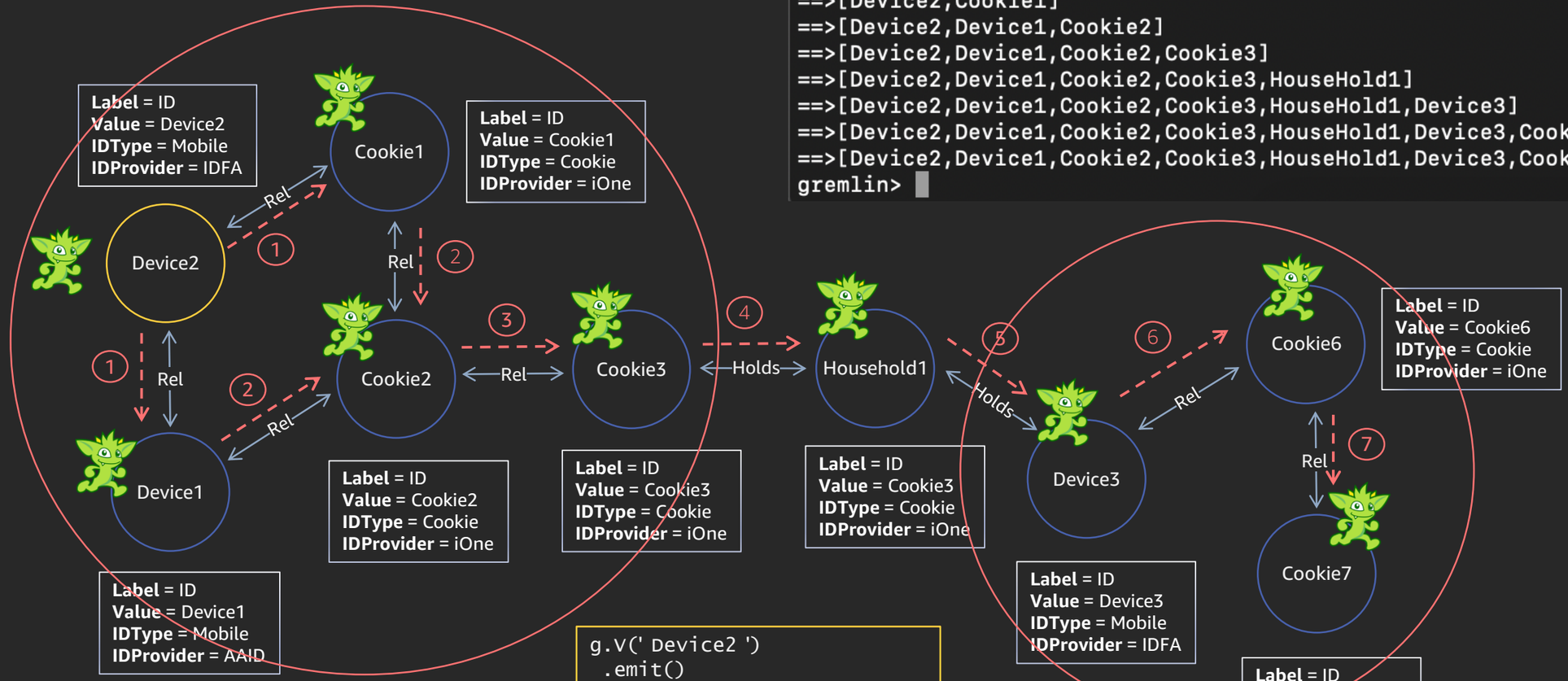


Apache TinkerPop Gremlin

Household graph sample



```
gremlin> g.V(1).emit().repeat(both().dedup()).dedup().path().by('name')
==>[Device2]
==>[Device2,Device1]
==>[Device2,Cookie1]
==>[Device2,Device1,Cookie2]
==>[Device2,Device1,Cookie2,Cookie3]
==>[Device2,Device1,Cookie2,Cookie3,HouseHold1]
==>[Device2,Device1,Cookie2,Cookie3,HouseHold1,Device3]
==>[Device2,Device1,Cookie2,Cookie3,HouseHold1,Device3,Cookie6]
==>[Device2,Device1,Cookie2,Cookie3,HouseHold1,Device3,Cookie6,Cookie7]
gremlin>
```




```
g.V(' Device2 ')
 .emit()
 .repeat(both().dedup())
 .dedup()
 .id()
 .toList();
```

Lessons learned: Neptune

Re-triable exceptions

Reader instance restarts

Batch deletes

DNS caching

Timeout per query

Key considerations: IDR

- Use TTL for data retention
- Mix of deterministic and probabilistic data in your graph
- Data security with encryption at rest and in transit
- Follow industry regulations and laws (GDPR & CCPA)
- Honoring opt-out requests

Thank you!

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws



Please complete the session survey in the mobile app.

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.