Angela
Scranton
Micheal
Located in
Managed by
Works in
Located in
Located in
ccounting
Dwight
Jim
Works in
Works in
ated in
Works in
Sales
Karen

Db2Night Show – September 25, 2020

Db2 Graph
Db2 REST

Agenda

Today we'll talk about two of the new features available in IBM Db2 11.5.4: a technical preview of **Db2 Graph** and the GA of **Db2 REST**.

01
Db2 Graph

02
Db2 REST

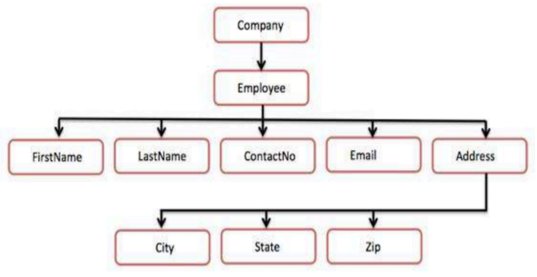
03
Demo

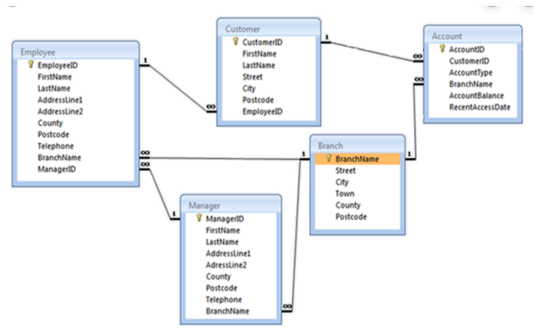
04
Getting started

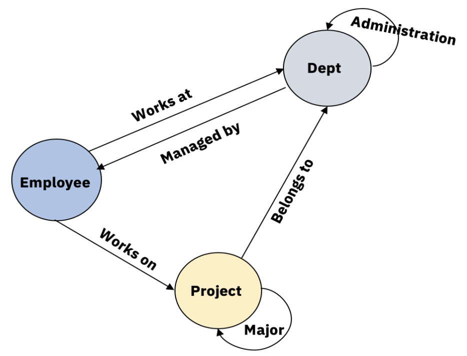
Graph databases are all around us

- Your favorite streaming service uses Graph databases to manage all their assets, to understand their users' viewing habits and to create personalized recommendations for movies and shows
- Companies manage the entire supply chain, inventory, orders and user history with Graph databases
- Your airline or hotel booking system may be using Graph databases to manage and recommend new options for users and build reports
- Your network provider may be using Graph databases to model networks to manage issues, like if a cell tower goes down or in case of a breach
- Your search engine or navigation system was built using Graph principles

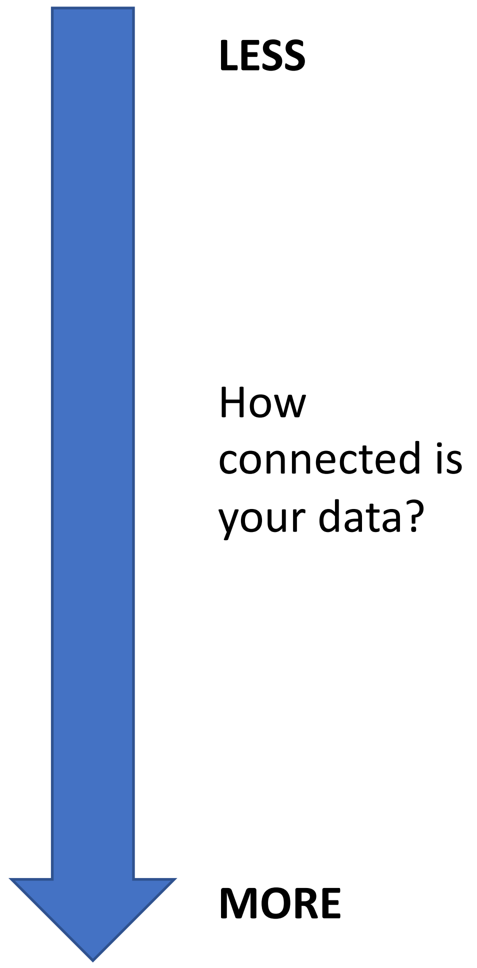
How is data stored?





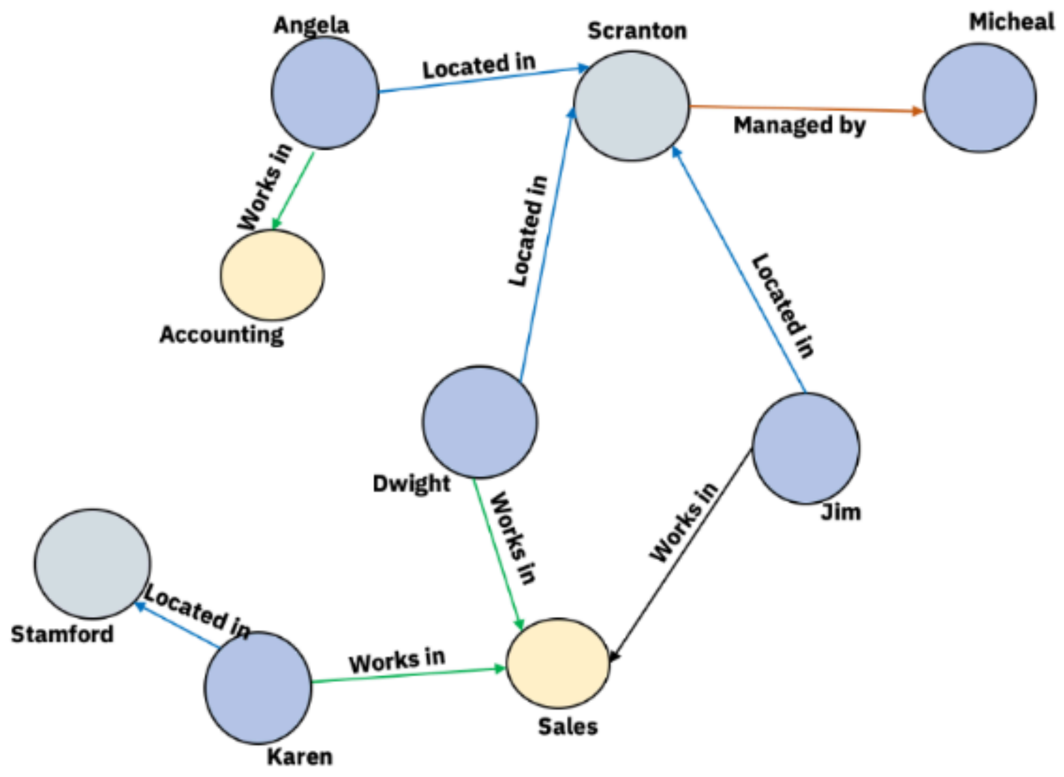


- **NoSQL** (document stores)
  - Stores sets of disconnected data, may be duplication, no ACID compliance.
  - Fast updates and retrievals of sets of data.
- **RDBMS**
  - Relationships defined by joins.
  - Good for transactions, aggregations, transformations.
  - Can’t handle indirect/complex relationships.
- **Graph**
  - Traverses the network to find indirect relationships and patterns.
  - Intuitive to query and visualize from different starting points.
  - Can’t handle fast updates or aggregations.



What is a property graph?

- Based on objects called **vertices** and their relationships, called **edges**.
- Each vertex or edge has:
  - An **ID** that is unique across the graph
  - A **label** that represents the type of object
  - A set of **properties** that provide additional attributes.



Native vs Non-native graph databases

NATIVE:

- Specialized data store and query engine
- Query performance for Graph queries

EXAMPLES:

- Neo4j, OrientDB, TigerGraph

NON-NATIVE:

- Use existing data store or materializes Graph in-memory
- Uses a complex schema to support both uses
- Specialized query engine for Graph

EXAMPLES:

- MSSQL server graph extension, ArangoDB, JanusGraph

What delays adoption of existing solutions?

- **Native graph solutions** that are suitable for graph-only workloads require you to migrate data from existing systems.
- **Hybrid solutions** that duplicate data in-database or in-memory or represent data in complex formats that aren’t easily accessible.

- High setup and maintenance costs.
- Relational database are used to solve many graph use cases.
- Duplicate data in another database or in-memory.
  - Inconsistent copies of data.
  - Export/import overhead to load data.
  - Cost of additional storage.

What is Db2 Graph?

- An enterprise grade in-database Graph solution
  - No movement of data into secondary storage to perform graph queries
- No additional cost to existing Db2 licenses
- Queries data and relationships with existing relational tables, supporting Graph analytics and traditional SQL on the same data
- Fetches data in real-time through JDBC

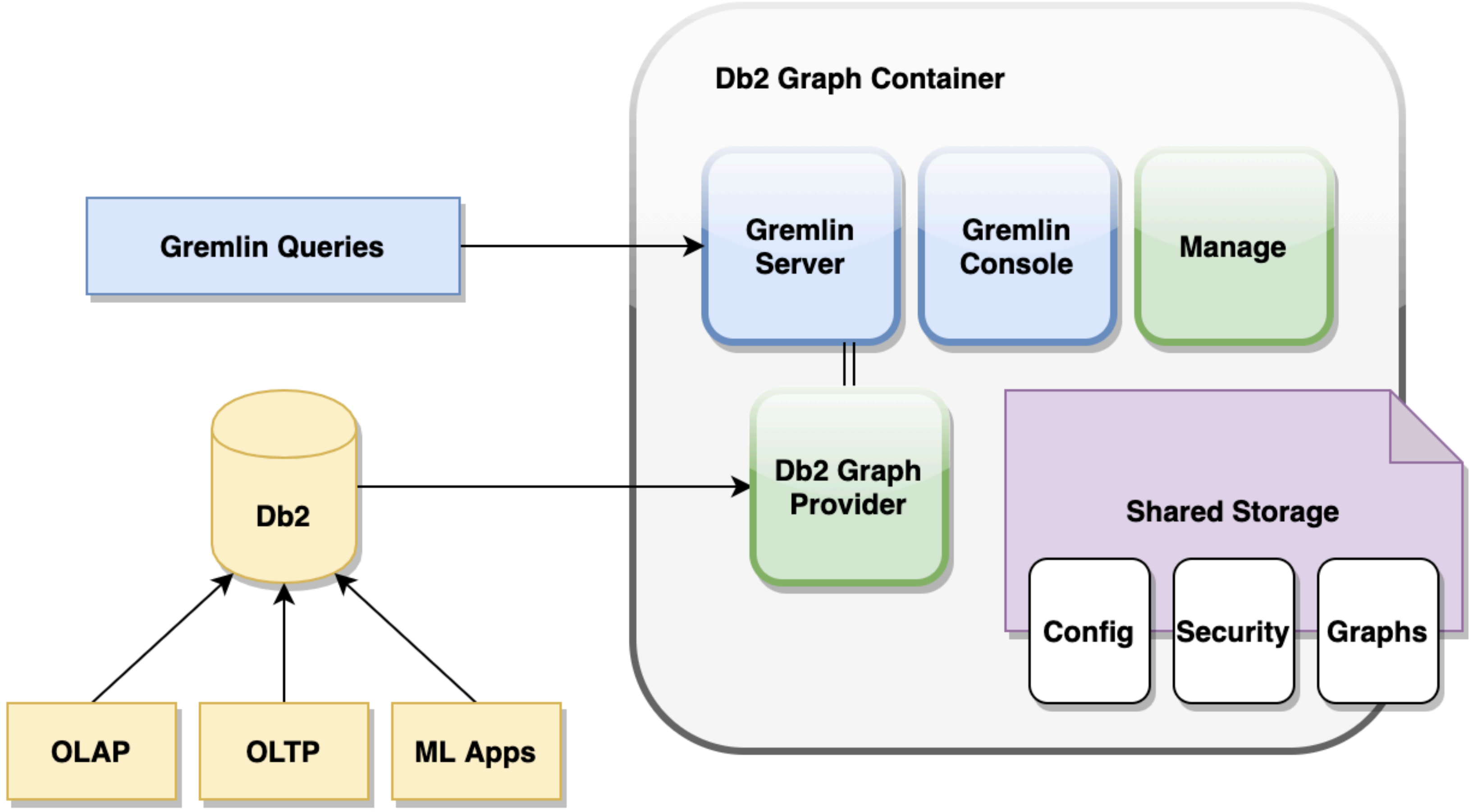
What is Db2 Graph?

- Based on Apache **TinkerPop**
  - TinkerPop is an open-source graph framework
  - **Gremlin** is the graph query language of TinkerPop
  - Db2 Graph is a provider plugin for TinkerPop
- Currently ships in a container that includes the Gremlin Server and Gremlin Console
- Released as technical preview in Db2 11.5.4

How does Graph differ from SQL?

Essential Operations on Data	RDBMS	GRAPH
Simple relationships between records	●	
Transaction information	●	
Summation and max queries	●	
Business intelligence (aggregations)	●	
Complex queries on interrelated data		●
Deep traversals		●
Indirect relationships (friend of a friend) k-hop traversals		●
Expressive query language and visual results		●
Versioning or auditing of data	●	
Fast execution of complex pattern matching queries		●
Represent multiple versions of the same graph		Db2 Graph

Db2 Graph Components



How does Db2 Graph work?

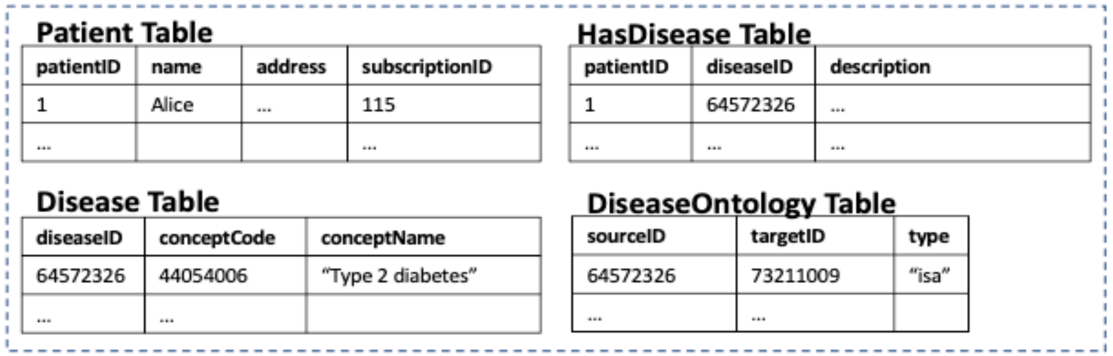
- Defines a virtual graph model, the **topology**, on top of Db2 tables
  - Uses referential information to automatically create a graph schema
  - Maps tables or views to graph vertices and edges
- Users create a **topology** using the container’s manage command
- Db2 Graph uses the topology to convert Gremlin queries to SQL

How does it work?

- Db2 remains untouched. No change in data, structure or performance.
- Existing applications are uninterrupted.
- Data is fetched by Db2 Graph at time of execution. Data consistency and updates are reflected in real-time.

Mapping relational data to a graph model

- Persists the identification of vertices and edges in a json model.
  - Vertices represent tables with a primary key
  - Edges represent relationships between tables identified by a foreign key
- Views are supported, but not included in the auto-generation and must be manually added.



Patient Table

patientID	name	address	subscriptionID
1	Alice	...	115
...			...

HasDisease Table

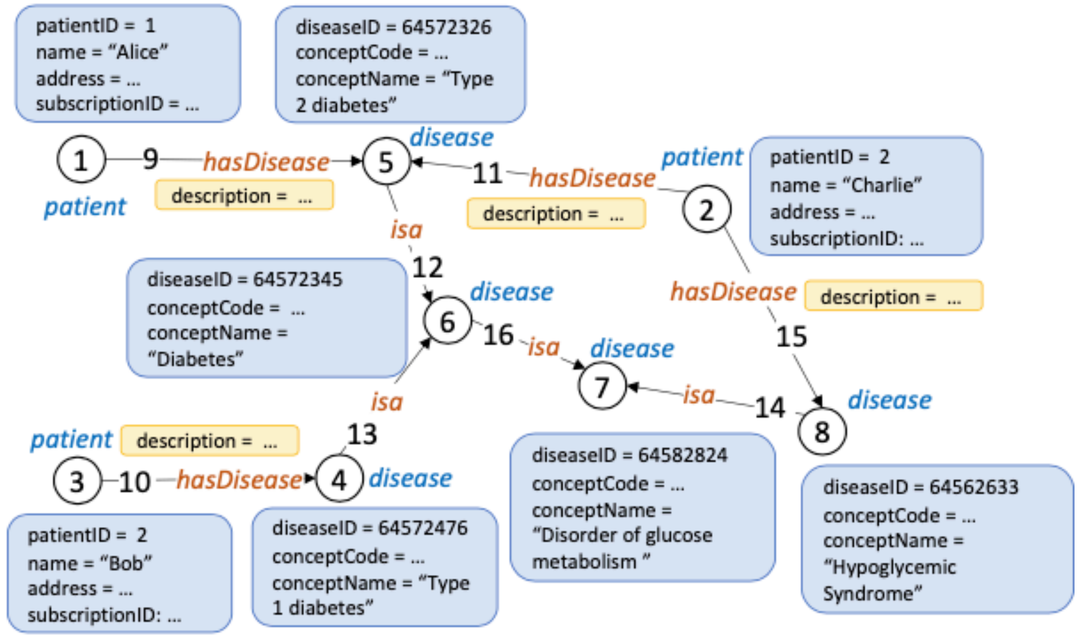
patientID	diseaseID	description
1	64572326	...
...	...	...

Disease Table

diseaseID	conceptCode	conceptName
64572326	44054006	"Type 2 diabetes"
...	...	

DiseaseOntology Table

sourceID	targetID	type
64572326	73211009	"isa"
...	...	



patientID = 1 name = "Alice" address = ... subscriptionID = ...
diseaseID = 64572326 conceptCode = ... conceptName = "Type 2 diabetes"
1
9
hasDisease
description = ...
5
disease
patient
11
hasDisease
description = ...
patient
2
patientID = 2 name = "Charlie" address = ... subscriptionID: ...
isa
12
diseaseID = 64572345 conceptCode = ... conceptName = "Diabetes"
6
disease
hasDisease
description = ...
16
isa
disease
15
7
isa
14
disease
8
isa
13
patient
description = ...
3
10
hasDisease
4
disease
patientID = 2 name = "Bob" address = ... subscriptionID: ...
diseaseID = 64572476 conceptCode = ... conceptName = "Type 1 diabetes"
diseaseID = 64582824 conceptCode = ... conceptName = "Disorder of glucose metabolism "
diseaseID = 64562633 conceptCode = ... conceptName = "Hypoglycemic Syndrome"

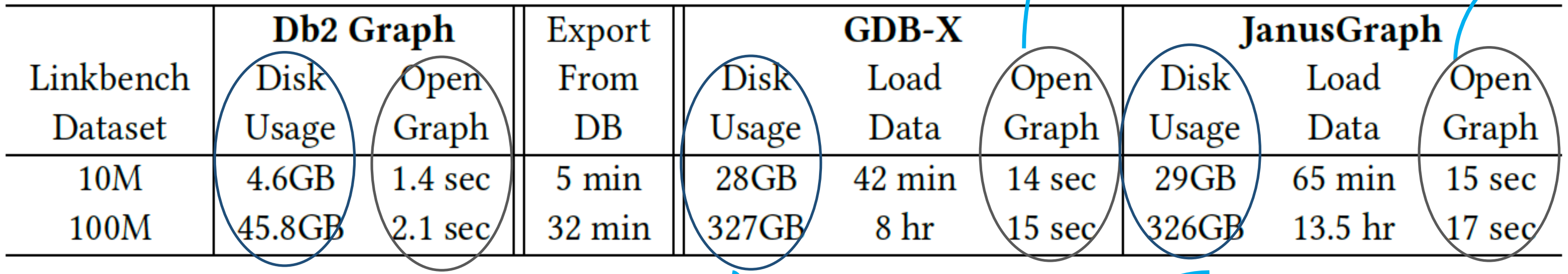
How does it perform?

- LinkBench graph & queries
  - 10 million node and 100 million node sample graphs
  - 32 core CPU, 256GB memory
  - Db2 and Db2 Graph running on same machine

LinkBench Query	Gremlin
getNode(id, lbl)	g.V(id).hasLabel(lbl)
countLinks(id1, lbl)	g.V(id1).outE(lbl).count()
getLink(id1, lbl, id2)	g.V(id1).outE(lbl).filter(outV().id() == id2)
getLinkList(id1, lbl)	g.V(id1).outE(lbl)

How does it perform?

Opening the graph takes ~9x longer, because of aggressive caching or complex data formats

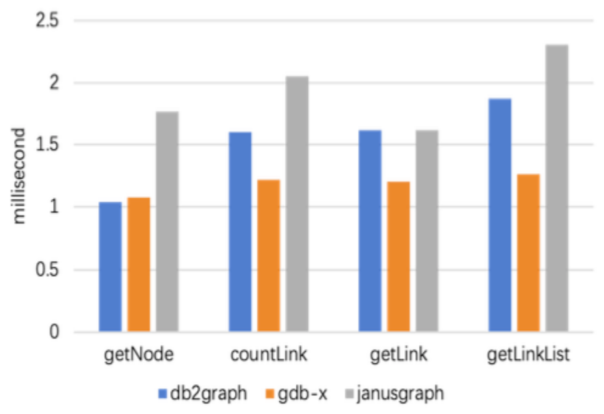


	Db2 Graph		Export	GDB-X			JanusGraph		
Linkbench Dataset	Disk Usage	Open Graph	From DB	Disk Usage	Load Data	Open Graph	Disk Usage	Load Data	Open Graph
10M	4.6GB	1.4 sec	5 min	28GB	42 min	14 sec	29GB	65 min	15 sec
100M	45.8GB	2.1 sec	32 min	327GB	8 hr	15 sec	326GB	13.5 hr	17 sec

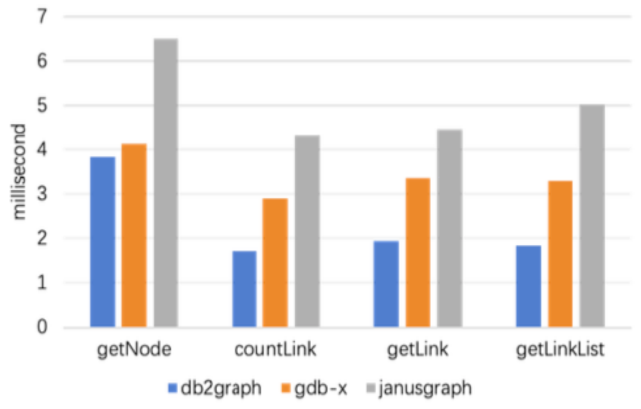
Need 6-7x more space to store data in the format

How does it perform?

- Latency – lower is better



(a) Linkbench-10M



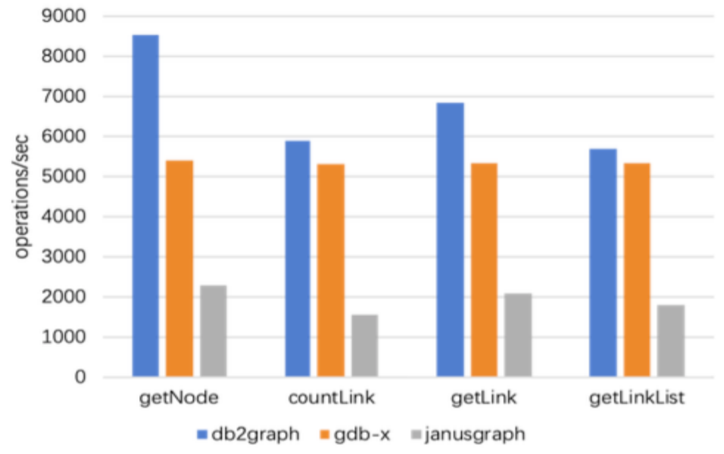
(b) Linkbench-100M

- gdb-x performs well for smaller graphs because of their caching mechanism

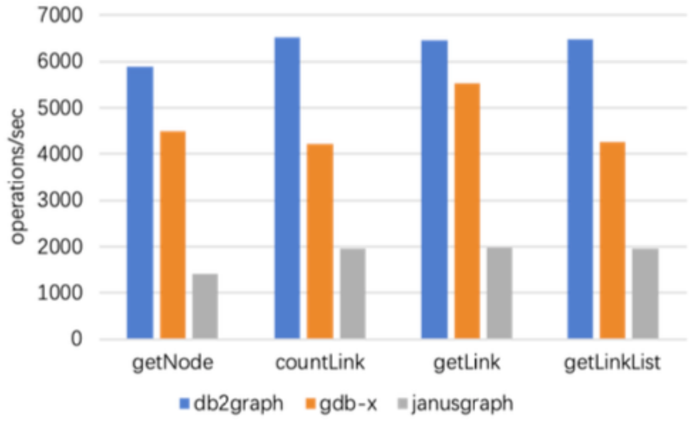
- gdb-x does not perform well when the size increase. For Db2 Graph, the data fits in Db2’s buffer pool

How does it perform?

- Throughput – higher is better



(a) Linkbench-10M



(b) Linkbench-100M

- Db2 Graph is the clear winner, the underlying Db2 engine is good at handling concurrent queries.

Db2 REST

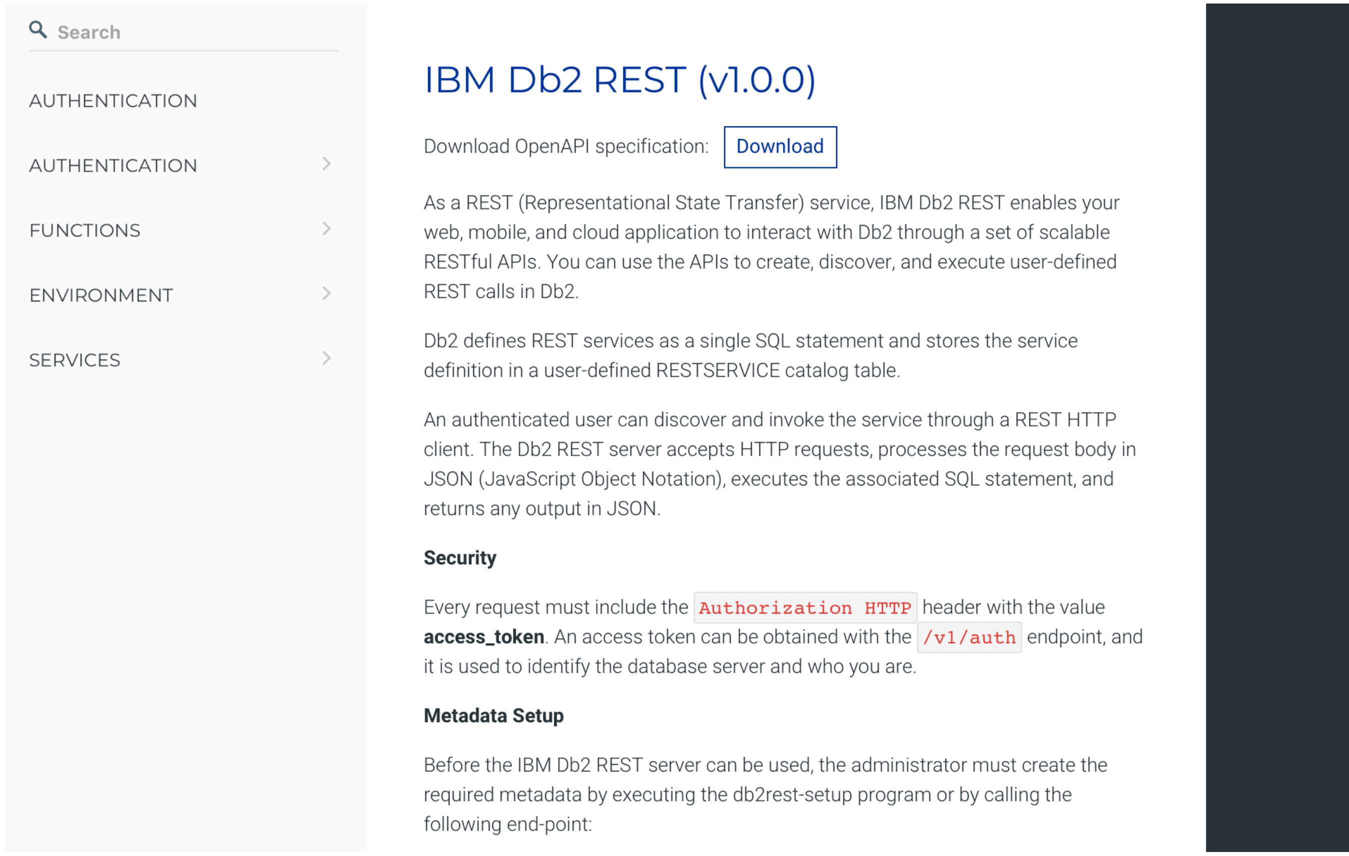
- Build and maintain apps easily without worrying about the deployment specifics or language dependencies, gives flexibility in using application development technologies.
  - IBM Db2 REST allows your web, mobile and cloud application to interact with Db2 through a set of scalable RESTful APIs.
  - Provides the application developer with APIs to **create**, **discover** and **execute** their own end-points, referred to as services.
  - Each **developer-defined service** is associated with a single SQL statement.
  - Services can be executed synchronously or as a job that supports result paging.

Db2 REST

- Released as part of Db2 11.5.4 in June as a standalone (Linux) container on IBM Cloud Container Registry.
- Available on IBM Integrated Analytics System and Db2 Warehouse (local deployments) since February.

Db2 REST: On-line Documentation

- Once the Db2 REST server is running, on-line documentation for each of the REST end-points is available at:
  - http://hostname:50050/docs



IBM Db2 REST (v1.0.0)

Download OpenAPI specification: Download

As a REST (Representational State Transfer) service, IBM Db2 REST enables your web, mobile, and cloud application to interact with Db2 through a set of scalable RESTful APIs. You can use the APIs to create, discover, and execute user-defined REST calls in Db2.

Db2 defines REST services as a single SQL statement and stores the service definition in a user-defined RESTSERVICE catalog table.

An authenticated user can discover and invoke the service through a REST HTTP client. The Db2 REST server accepts HTTP requests, processes the request body in JSON (JavaScript Object Notation), executes the associated SQL statement, and returns any output in JSON.

Security

Every request must include the `Authorization HTTP` header with the value **access_token**. An access token can be obtained with the `/v1/auth` endpoint, and it is used to identify the database server and who you are.

Metadata Setup

Before the IBM Db2 REST server can be used, the administrator must create the required metadata by executing the db2rest-setup program or by calling the following end-point:

Running the Db2 REST server

- Use scripts to operate the REST server.
  - docker exec containername /opt/ibm/dbrest/scripts/scriptname

Script	Description
db2rest-start.sh	Starts the server and changes its status to ACTIVE.
db2rest-stop.sh	Stops the server and changes it status to INACTIVE. Any authentication tokens are automatically invalidated.
db2rest-restart.sh	Stops and restarts the server.
db2rest-version.sh	Returns the version information of a running server.
db2rest-status.sh	Checks the status (ACTIVE, INACTIVE, ERROR) of the server.

Initializing REST services

- Db2 REST stores metadata in the Db2 instance.
- Metadata is created post-docker run using:
  - db2rest-setup.sh
  - A REST call to https://hostname:50050/v1/metadata/setup
- All management of services should be performed through REST calls and not by direct manipulation of the metadata table.

REST features

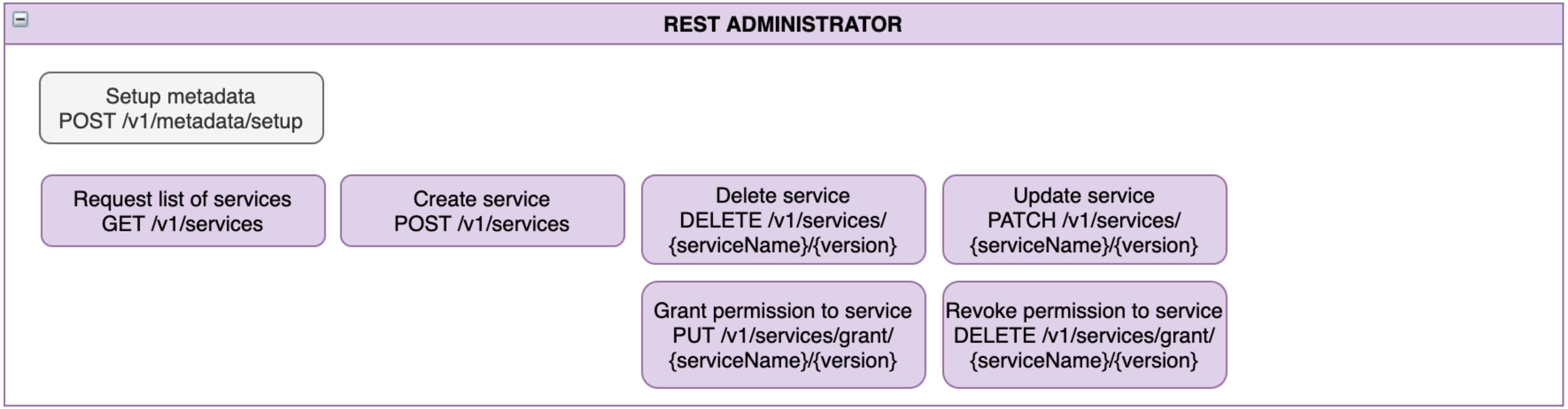
- Db2 REST is deployed as a docker container supporting Db2 11.1 and Db2 11.5.
- Authenticated users can use these REST end-points from any REST HTTP client without installing Db2 drivers.
- The Db2 REST server accepts an HTTP request, processes the request body and returns results in JSON.

REST features

- Main functions include:
    - Request authentication token
    - Create REST service end-point
    - Update/delete REST end-points
    - List and describe available end-points
    - Execute and monitor job process
    - Set service creation and execution permissions

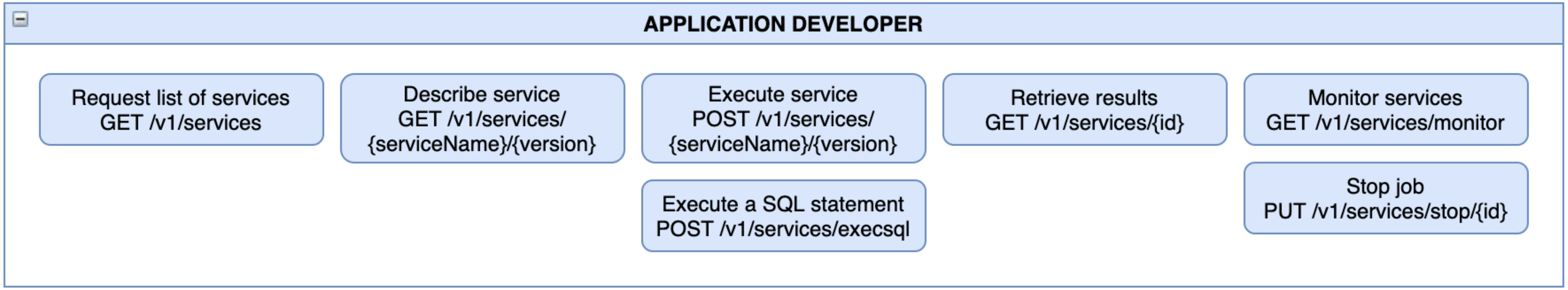
REST administration end-points

- REST end-points are split between actions performed by an administrator of REST and application developer who uses the end-points.


REST ADMINISTRATOR
Setup metadata
POST /v1/metadata/setup
Request list of services
GET /v1/services
Create service
POST /v1/services
Delete service
DELETE /v1/services/{serviceName}/{version}
Update service
PATCH /v1/services/{serviceName}/{version}
Grant permission to service
PUT /v1/services/grant/{serviceName}/{version}
Revoke permission to service
DELETE /v1/services/grant/{serviceName}/{version}

REST application end-points

- REST end-points to discover, execute and monitor services.


APPLICATION DEVELOPER
Request list of services GET /v1/services
Describe service GET /v1/services/{serviceName}/{version}
Execute service POST /v1/services/{serviceName}/{version}
Execute a SQL statement POST /v1/services/execsql
Retrieve results GET /v1/services/{id}
Monitor services GET /v1/services/monitor
Stop job PUT /v1/services/stop/{id}

DEMO

Getting started

- The technology preview of IBM Db2 Graph and GA of IBM Db2 REST are both available as docker containers hosted on ICR.

- To access Db2 Graph visit:
  - https://supportcontent.ibm.com/support/pages/technology-preview-ibm-db2-graph

- To access Db2 REST visit:
  - https://www.ibm.com/support/knowledgecenter/en/SSEPGG_11.5.0/com.ibm.db2.luw.admin.rest.doc/doc/c_rest.html