CptS 415 Big Data

# Labeled Property Graph

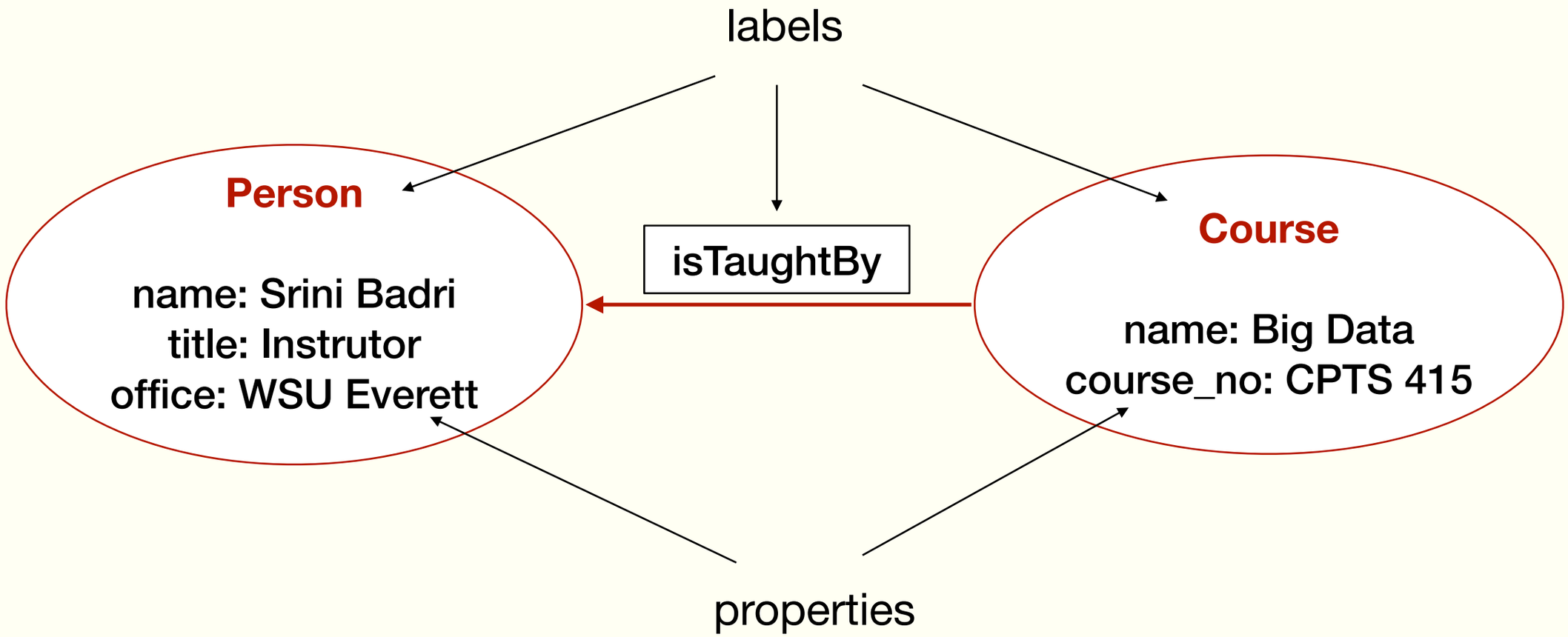
Srini Badri

# Labeled Property Graph (or Property Graphs)

- Vertices:
  - Vertices consist of one or more labels
  - Vertices also consist of one or more properties
  - Properties are defined using key-value pairs

- Edges:
  - Edges connect vertices
  - Edges consist of one or more labels
  - Edges also consist of one or more properties
  - Properties are defined using key-value pairs

# Example: Labeled Property Graph



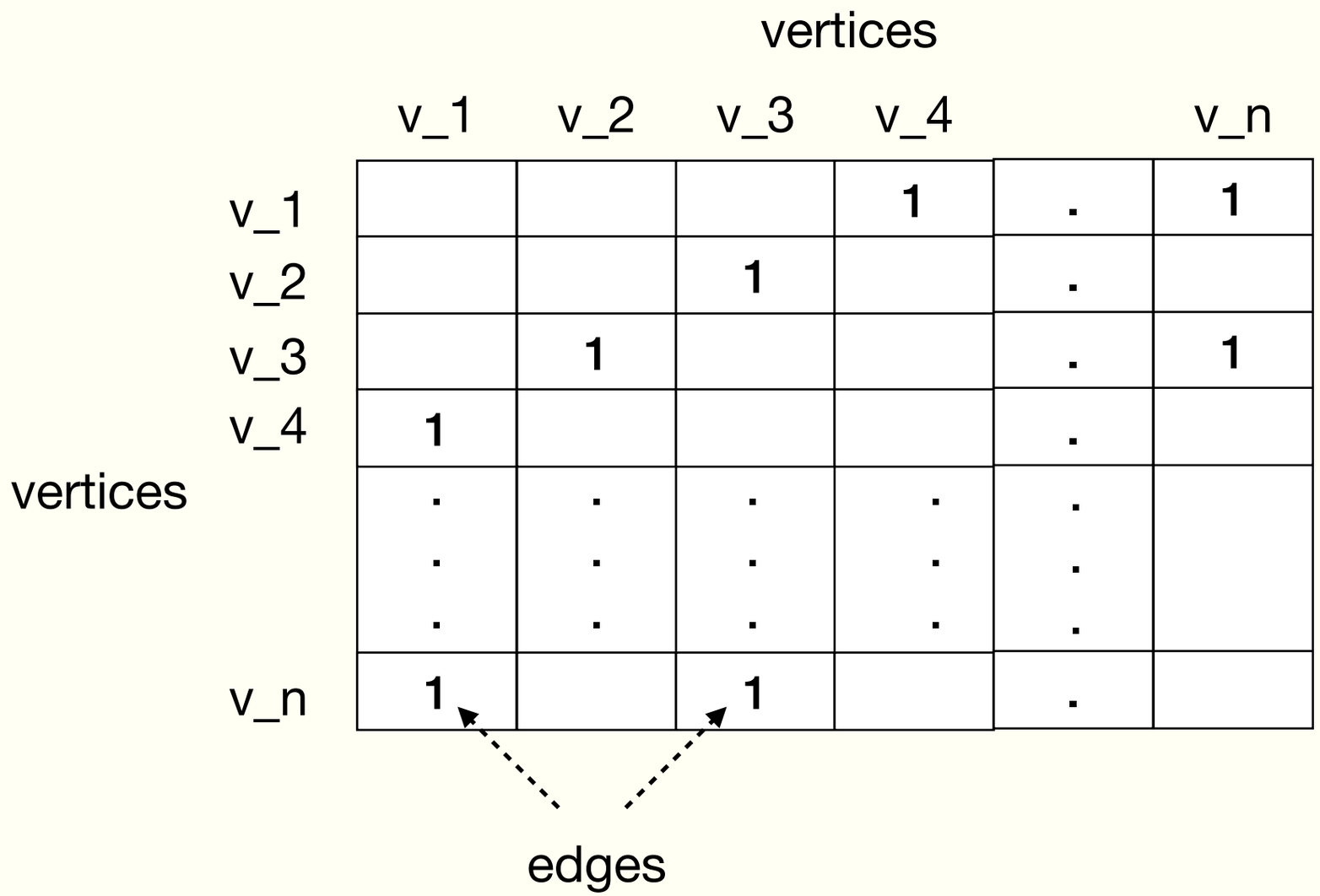
# Data Storage Schemes

- Data Storage Schemes:
  - native graph stores
  - key-value stores
  - tuples (similar to RDF)
  - document store (eg. JSON)
  - tables in RDBMS
  - object in OODBMS

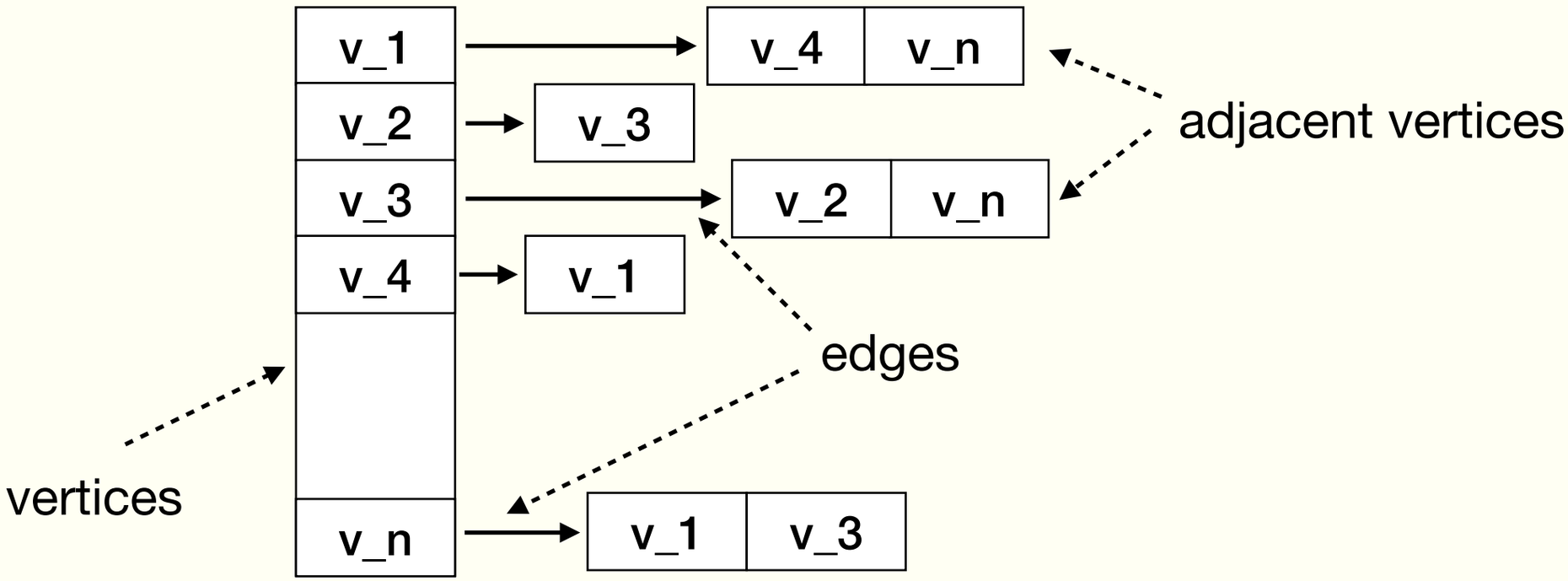
# Graph Databases

- Common Graph Databases
  - Neo4j (native graph store)
  - Titan, JanusGraph (wide-column store)
  - OrientDB (document store)
  - Azure Cosmos DB (document store)
  - MS Graph Engine (key-value store)
  - Nebula Graph (key-value store)
  - SAP HANA (column RDBMS)
  - Oracle (row RDBMS)
  - Mark Logic (document + RDF + table store)

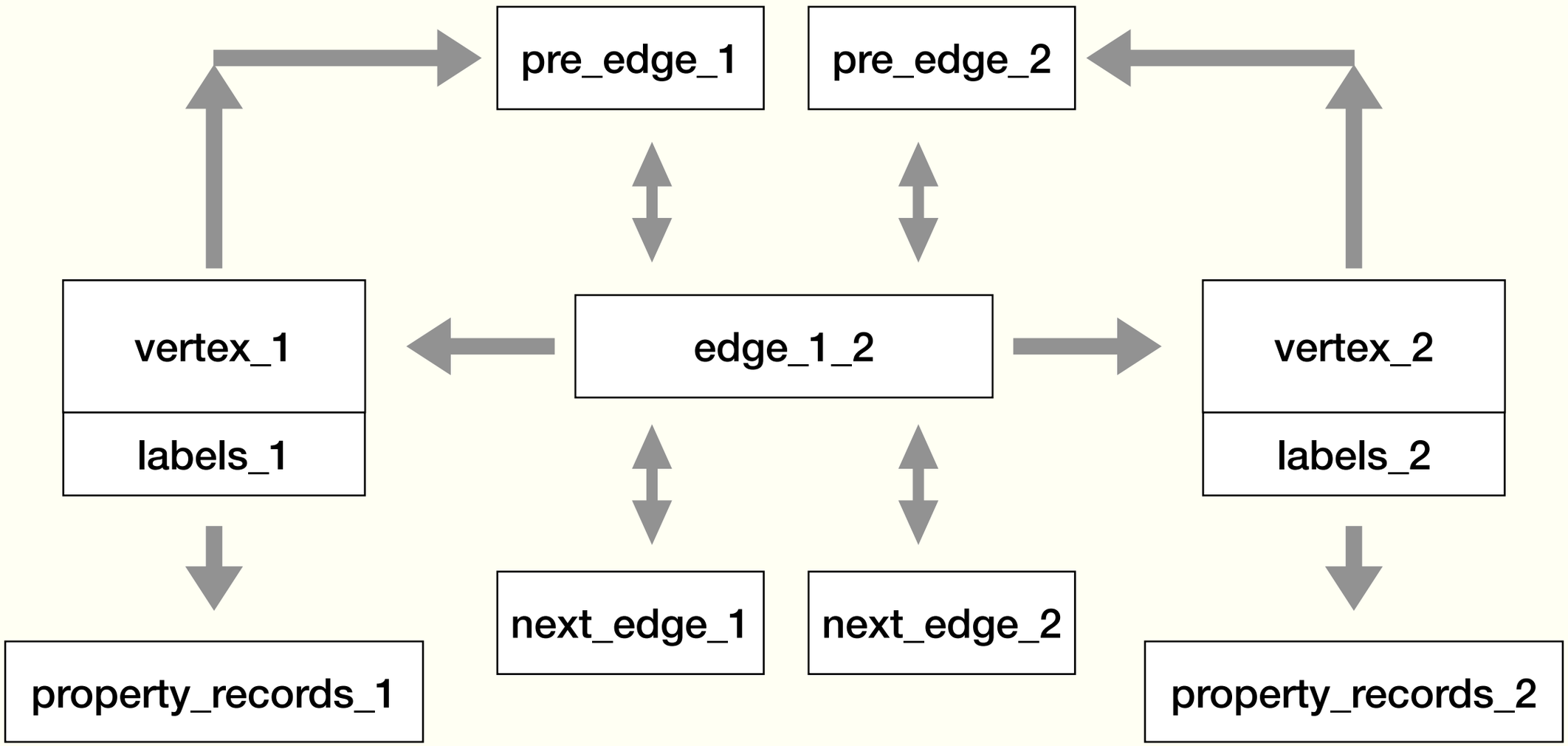
# Data Modeling - Adjacency Matrix



# Data Modeling - Adjacency List



# Neo4j - Graph Model



# Query Languages:

- Cypher
  - Neo4j graph query language

- Gremlin:
  - Graph traversal language

- PGQL:
  - SQL-like query language for Oracle graph database

- GQL:
  - Graph query language standard - work in progress

# Neo4j - Schema

- Constraints:
  - Node Property Uniqueness
  - Node Property Existence (not null)
  - Edge Property Existence (not null)
  - Node Key Constraint (labels and properties)

- Index
  - Single property index
  - Composite property index
  - Relationship type lookup index
  - Node label lookup index

# Neo4j - Cypher CREATE Statements

```
CREATE (Student_001:Student {ID: "001", name: "Joe"})
CREATE (Student_002:Student {ID: "002", name: "Mary"})
CREATE (Course_331:Course {Course_No:"331", title:"DB", credit:3.0})
CREATE (Course_350:Course {Course_No:"350", title:"Web", credit: 4.0})
CREATE
(Student_001)-[:Enrolled_In {grade:"B+"}]->(Course_331),
(Student_002)-[:Enrolled_In {grade:"A"}]->(Course_331)
CREATE
(Student_002)-[:Enrolled_In {grade:"A"}]->(Course_350)
```

# Neo4j - Cypher CREATE CONSTRAINT Statements

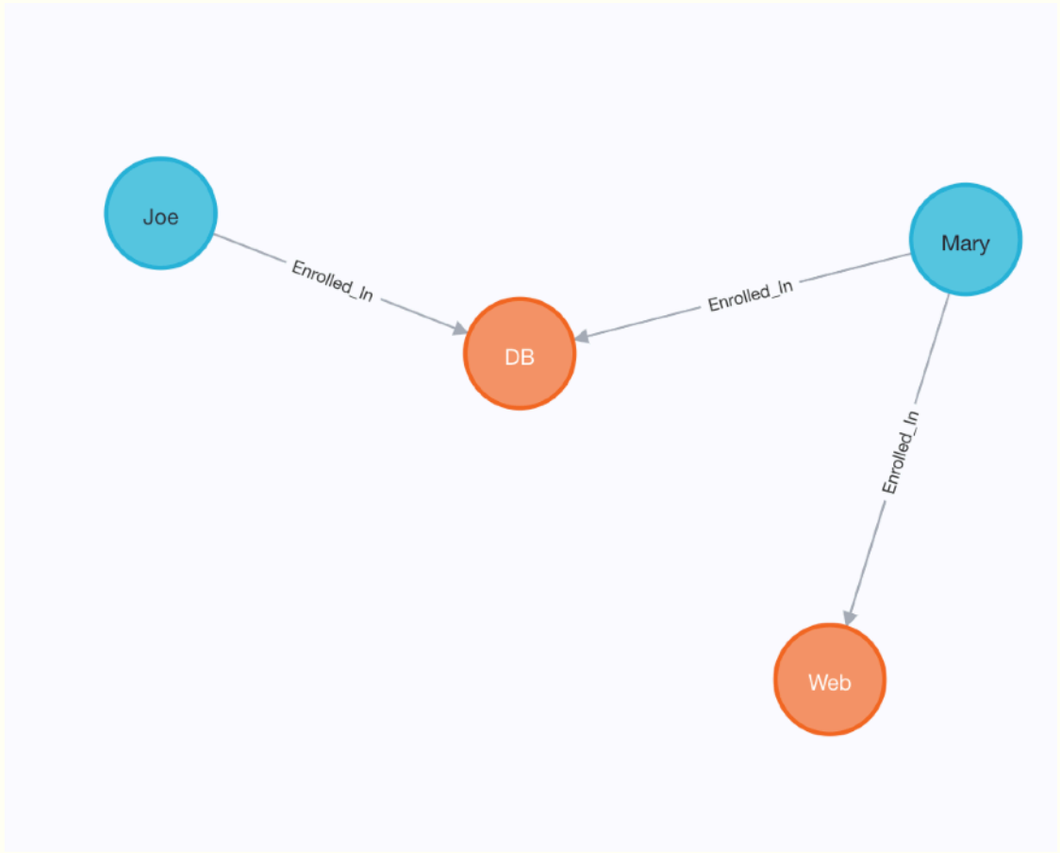
```
CREATE CONSTRAINT Student_ID_Constraint ON (s:Student)
ASSERT s.ID IS UNIQUE;
```

```
CREATE CONSTRAINT Course_No_Constraint ON (c:Course)
ASSERT c.Course_No IS UNIQUE;
```

```
CREATE CONSTRAINT Student_Name_Constraint ON
(s:Student) ASSERT s.name IS NOT NULL;
```

# Neo4j - Query and Visualization

```
MATCH (n) RETURN n
```



# Neo4j - Query and Visualization

```
MATCH (s:Student)-[r:Enrolled_In]->(c:Course) RETURN s.ID
AS STUDENT_ID, c.Course_No AS COURSE_NO, r.grade AS
GRADE
```

| STUDENT_ID | COURSE_NO | GRADE |
|---|---|---|
| "002" | "331" | "A" |
| "001" | "331" | "B+" |
| "002" | "350" | "A" |

# Neo4j - Cypher DELETE Statements

```
MATCH (n)
DETACH DELETE n
```

```
MATCH (n {name: 'Joe'})
DETACH DELETE n
```

# Neo4j - API Support

- Drivers:
  - Neo4j provides drivers for different programming languages
  - Supported languages:
    - Java, Python, C/C++, JavaScript, R, and others..

- Resources:
  - https://neo4j.com/developer/language-guides/
  - https://neo4j.com/docs/api/python-driver/current/index.html

# Neo4j - Python Example

```
from neo4j import GraphDatabase

def main():
    db = GraphDatabase.driver("bolt://localhost:7687", auth=("neo4j", "password"))
    session = db.session(database="neo4j")
    results = session.run("MATCH (n) RETURN n")
    for record in results:
        print(record)
    session.close()
    db.close()

if __name__ == "__main__":
    main()
```

# Summary

- Labeled Property Graph or Property Graph is a widely used graph model

- Labeled Property Graph involves:
  - Labels for Vertices and Edges
  - Properties for Vertices and Edges

- Labeled Property Graph are implemented differently by different databases:
  - Native graph implementation, key-value store based implementation, document store based implementation, etc.

- Neo4j is a popular graph database that uses native graph implementation and provides Cypher query language