W3. Lecture Notes — By Junyi

1•Common considerations
•Choice of classes and relations
•Reification and handling higher arity relationships
2•RDF knowledge graph design driven by
•Use over the WWW
•Reuse of vocabularies
•Linking data
3•Property Graph design driven by
•Optimizing query performance
Even though there are some guidelines to design, there are also equivalent good choices.

Outline - design schema

- ▼ design RDF graph
  - linked data principles
    - ▼ use IRI as names of things
      - identify the items of interest
      - short and pneumonic
      - ensure persistence
    - ▼ Use HTTP IRIs
      - people can lookup (dereferencing) things
    - ▼ provide useful information in RDF/SPARQL
      - dereferencing of a non-information object returns RDF data
      - if must create new vocabulary, ensure...
        - documented, self-describing, visioning policy, defined in multiple languages, published by trusted sources
    - ▼ Include links to other URIs

        - relationship links
            - <foaf:based_near>
            - relate object in one dataset to an object in another
        - identity links
            - <owl:sameAs>
            - equate object in one dataset to objects in another
        - vocabulary links
            - <rfbs:subClassOf>
            - links from the data to the definition of terms

▼ design property graph

- ▼ choosing nodes, labels, and properties
- ▼ when to introduce relationships
    - when efficient access is required
- ▼ when to introduce a relationship property
    - common use cases for relationship properties
        - time varying relationships
        - provenance
        - confidence
    - disadvantages
        - Many systems do not index on relationship properties
        - This may not be a problem if relationship properties are used in the last stage of query processing
        - For performance sensitive queries, it is better to reify the relationship