UNIVERSITETET
I OSLO

Semistrukturerte data og XML

Where is the Life we have lost in living?
Where is the wisdom we have lost in knowledge?
Where is the knowledge we have lost in information?

-- T. S. Eliot

Where is all the information we have lost in data structures?

-- Albrecht Schmidt, Universitetet i Ålborg

Semistrukturerte data – bakgrunn

- Dataintegrasjon
  - forskjellige, uavhengige datakilder
  - forskjellige datamodeller og skjemaer
- Kunnskapsrepresentasjon og informasjonsutveksling
  - Semantic Web
  - Metainformasjon som del av dataene
- Presentasjon av data på weben
- Ekstrahering av data fra weben

Semistrukturerte data

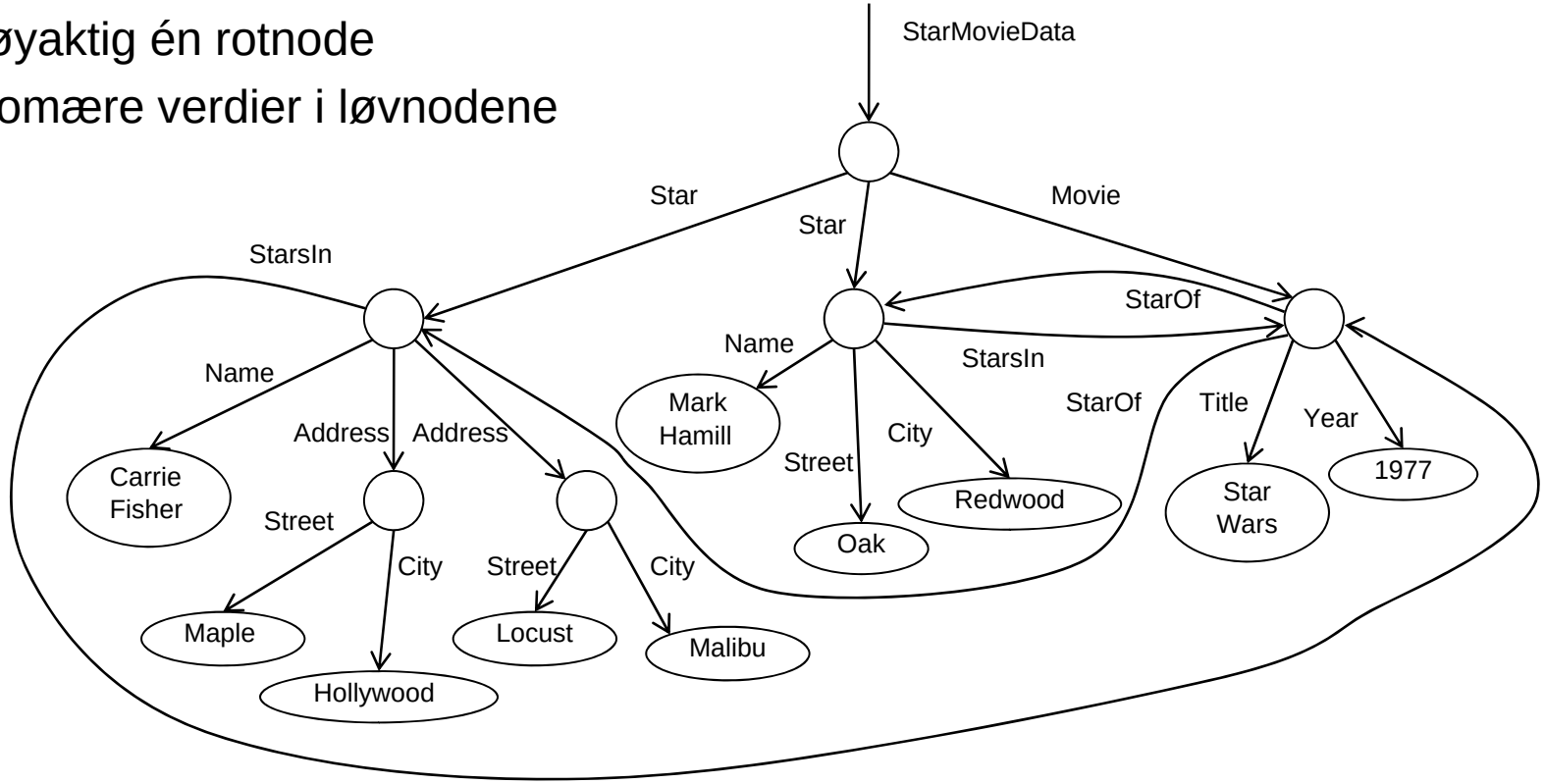
- Strukturerte data
  - Tabeller i relasjonsdatabaser, ...
- Ustrukturerte data
  - Tekstdokumenter
- Semistrukturerte data:
  Deler har fast format, deler består av fri tekst
  - Bøker: Tittelark (tittel, forfatter, utgivelsesår, ©, ISBN, ..), kapitteloverskrifter, ... , tekstavsnitt
  - Websider: I utgangspunktet utformet for å leses og tolkes av mennesker, men med formatering og annen maskintolkbar informasjon på deler av innholdet
  - CVer: Navn, fødselsdato, utdannelse, arbeidserfaring, ... , prosjektbeskrivelser, forskningsplan, ...

XML

- XML – eXtensible Markup Language:
  - Språk for å beskrive skjema- og annen metainformasjon som del av dataene
  - Kan brukes til å beskrive semistrukturerte data
- I XML-porteføljen:
  - Skjemaspråk: DTD, XML Schema
  - Spørrespråk: XPath, XQuery
  - Transformasjonsspråk: XSLT, CSS
  - APIer: DOM, SAX
  - ...

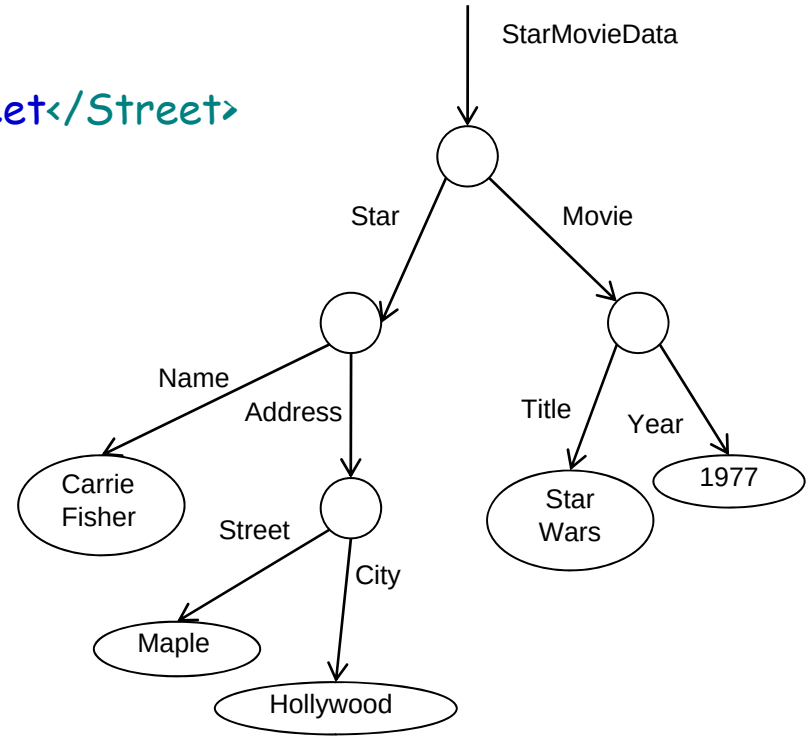
Grafrepresentasjon

- Rettet graf (treaktig) med navngitte kanter
  - Kantene representerer relasjoner og attributter
- Nøyaktig én rotnode
- Atomære verdier i løvnodene



XML-eksempel

```
<?xml version="1.0" encoding="utf-8"?>
<StarMovieData>
    <Star>
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple Street</Street>
            <City>Hollywood</City>
        </Address>
    </Star>
    <Movie>
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
</StarMovieData>
```



Attributter

```
<?xml version="1.0" encoding="utf-8"?>
<StarMovieData>
    <Star starID="cf" starredIn="sw">
        <Name>Carrie Fisher</Name>
        <Address>
            <Street>123 Maple Street</Street>
            <City>Hollywood</City>
        </Address>
    </Star>
    <Movie movieID="sw" starOf="cf">
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>
</StarMovieData>
```

Navnerom

- For å lette gjenbruk av vokabularer
  - kan referere til flere vokabularer i samme XML-dokument uten å få navnekonflikter
  - navnerommet angis med en URI
    - URIen er typisk en URL som refererer til et dokument som beskriver tolkningen av taggene i navnerommet
    - dokumentet kan være et XML-dokument, en uformell beskrivelse, ... , eller ingenting
  - kan kvalifisere taggene i XML-dokumentet med navnerommet

Eksempel på bruk av navnerom

```
<?xml version = "1.0" encoding="utf-8"?>
<dmms:minFilmDB xmlns:md="http://infolab.stanford.edu/movies"
                xmlns:dmms="http://www.ifi.uio.no/dmms/filmer">
  <md:StarMovieData>
    <md:Movie>
        <md:Title>Star Wars</md:Title>
        <md:Year>1977</md:Year>
        <dmms:kommentarer> ... </dmms:kommentarer>
    </md:Movie>
  </md:StarMovieData>
  <dmms:Movie>
    ...
  </dmms:Movie>
</dmms:minFilmDB>
```

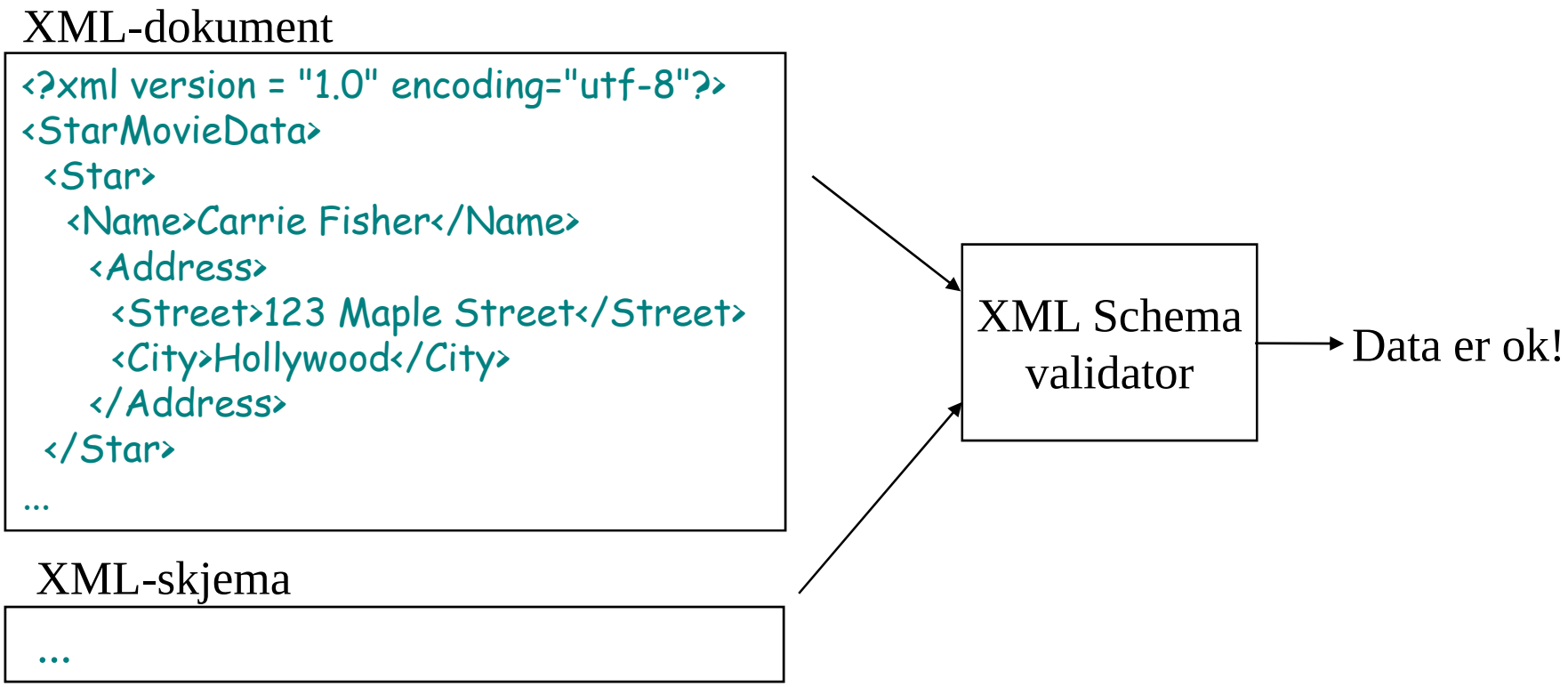
Velformede og gyldige XML-dokumenter

- **Velformet** XML-dokument
  - Dokumentet angir innledningsvis at det er et XML-dokument
  - Én rot
  - Korrekt XML-syntaks (parentetisk struktur på taggene, ...)

```
<?xml version = "1.0" encoding="utf-8"?>
 <StarMovieData>
   ...
 </StarMovieData>
```

- **Gyldig** (**valid**) XML-dokument
  - Elementene i dokumentet følger et gitt **skjema**

Validering av XML-dokumenter



XML Schema

```
<?xml version = "1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

</xs:schema>
```

Elementer

```
<?xml version = "1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <xs:element name="Title" type="xs:string" />
    <xs:element name="Year" type="xs:integer" />

</xs:schema>
```

Sammensatte typer

```
<?xml version = "1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:complexType name="movieType">


            <xs:sequence>
                     <xs:element name="Title" type="xs:string" />
                     <xs:element name="Year" type="xs:integer" />
            </xs:sequence>
   </xs:complexType>
   <xs:element name="Movies">
            <xs:complexType>
                     <xs:sequence>
                              <xs:element name="Movie" type="movieType"
                                   minOccurs="0" maxOccurs="unbounded" />
                     </xs:sequence>
            </xs:complexType>
   </xs:element>
</xs:schema>
```

Eksempeldokument

```
<?xml version = "1.0" encoding="utf-8"?>
<Movies xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="Movies.xsd">

    <Movie>
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>

    <Movie>
        ...
    </Movie>

    ...

</Movies>
```

Attributter

```
<?xml version = "1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xs:complexType name="movieType">
          <xs:attribute name="movieID" type="xs:string" use="required" />
          <xs:attribute name="starOf" type="xs:string" />
            <xs:sequence>
                    <xs:element name="Title" type="xs:string" />
                    <xs:element name="Year" type="xs:integer" />
            </xs:sequence>
   </xs:complexType>
   <xs:element name="Movies">
          <xs:complexType>
                    <xs:sequence>
                              <xs:element name="Movie" type="movieType"
                                   minOccurs="0" maxOccurs="unbounded" />
                    </xs:sequence>
          </xs:complexType>
   </xs:element>
</xs:schema>
```

Eksempeldokument

```
<?xml version = "1.0" encoding="utf-8"?>
<Movies xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:noNamespaceSchemaLocation="Movies.xsd">

    <Movie movieID="sw">
        <Title>Star Wars</Title>
        <Year>1977</Year>
    </Movie>

    <Movie movieID="rj">
        ...
    </Movie>

    ...

</Movies>
```

Nøkler

```
<?xml version = "1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
   ...
   <xs:element name="Movies">
            <xs:complexType>
                     <xs:sequence>
                              <xs:element name="Movie" type="movieType"
                                   minOccurs="0" maxOccurs="unbounded" />
                     </xs:sequence>
            </xs:complexType>
            <xs:key name="movieKey">
                     <xs:selector xpath="Movie" />
                     <xs:field xpath="Title" />
            </xs:key>
   </xs:element>

</xs:schema>
```

Fremmednøkler

```
…
<xs:element name="Stars">
  <xs:complexType>
    …
    <xs:element name="StarredIn"  minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
        <xs:element name="title" type="xs:string" />
        <xs:element name="year" type="xs:integer" />
      </xs:complexType>
    </xs:element>
    …
  </xs:complexType>
  <xs:keyref name="movieRef" refers "movieKey">
    <xs:selector xpath="Star/StarredIn" />
    <xs:field xpath="Title" />
  </xs:keyref>
</xs:element>
```

”XML-data er selvforklarende” (???)

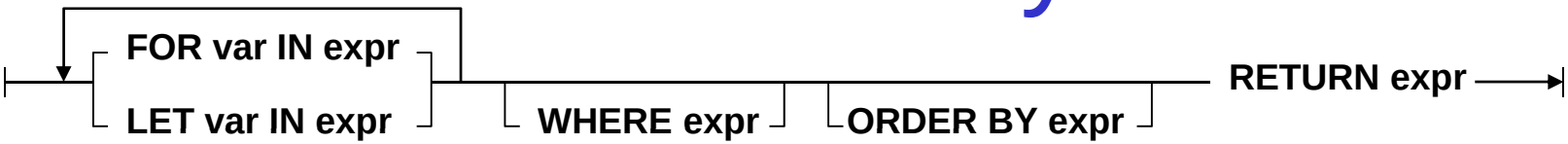
```
<?xml version = “1.0” encoding=”utf-8”?>
<外语>Китайська мова</外语>
```

- XML-taggene sier noe om dataenes struktur, men ikke noe om tolkningen
- XML-data er **selvdokumenterende**, men *ikke selvforklarende*...
  - men ved bruk av vokabularer/navnerom kan man indikere en tolkning

Xpath-uttrykk

- En sekvens av elementer kan angis ved hjelp av en absolutt eller relativ path av markeringer.
  - /Movies – rotelementet (med alt innhold).
  - /Movies/Movie – alle <Movie>-elementer innenfor <Movies>-elementet.
  - /Movies/Movie/Title – alle <Title>-elementer innenfor et <Movie>-element i <Movies>.
- //M angir alle <M>-elementer, uavhengig av nesting.
- * angir en vilkårlig markering
- Kan også angi betingelser som en del av uttrykket.
  - /Movies/Movie/[Year="1980"] – alle <Movie>-elementer hvor <Year>-elementet inneholder 1980.

FLWR-uttrykk



- FOR variable IN expression (, variable IN expression)*
- LET variable := expression (, variable := expression)*
- WHERE expression
- ORDER BY expression
- RETURN expression

FLWR-eksempler

- Eksempel 1:

```
let $a := (1, 2, 3)
return <out>{$a}</out>
```

- Eksempel 2:

```
for $a in (1, 2, 3)
return <out>{$a}</out>
```

Eksempel: XML-dokument

```
<?xml version = "1.0" encoding="utf-8"?>
<Movies>
        <Movie genre="comedy">
                <Title>Bruce Almighty</Title>
                <Star><Name>Jim Carrey</Name></Star>
        </Movie>
        <Movie genre="comedy">
                <Title>Dumb & Dumber</Title>
                <Star><Name>Jim Carrey</Name></Star>
        </Movie>
        <Movie genre="drama">
                <Title>The Truman Show</Title>
                <Star><Name>Jim Carrey</Name></Star>
        </Movie>
        <Movie genre="comedy">
                <Title>Nine Months</Title>
                <Star><Name>Hugh Grant<Name></Star>
        </Movie>
</Movies>
```

Eksempel: FLWR

Finn alle komedier med Jim Carrey som skuespiller:

```
let $movies := doc("movies.xml")
for $movie in
$movies//Movie[@Genre="comedy"]
where $movie/Star/[Name="Jim carrey"]
return $movie/Title
```

Andre muligheter i XQuery

- Join:

```
for $s1 in …, $s2 in …
where data($s1/…) = data($s2/…)
```

- Duplikat-eliminasjon:

```
let $s := distinct-values(…)
```

- Kvantorer:

```
every $s in … satisfies …
some $s in … satisfies …
```

- Aggregering (count, sum, max, …)
- Forgrening:

```
if (…) then … else …
```

Klasser av XML-dokumenter

- **Datasentrisk**:
  - Ganske regelmessig struktur
  - ”Finkornede ”data
- **Dokumentsentrisk**:
  - Irregulær struktur
  - Mer ”grovkornede” data
- **Hybrider** av disse:
  - Stort sett dokumentsentrisk, men med deler som er finkornede og med regelmessig struktur
  - Stort sett datasentrisk, men med deler som er grovkornede og med irregulær struktur

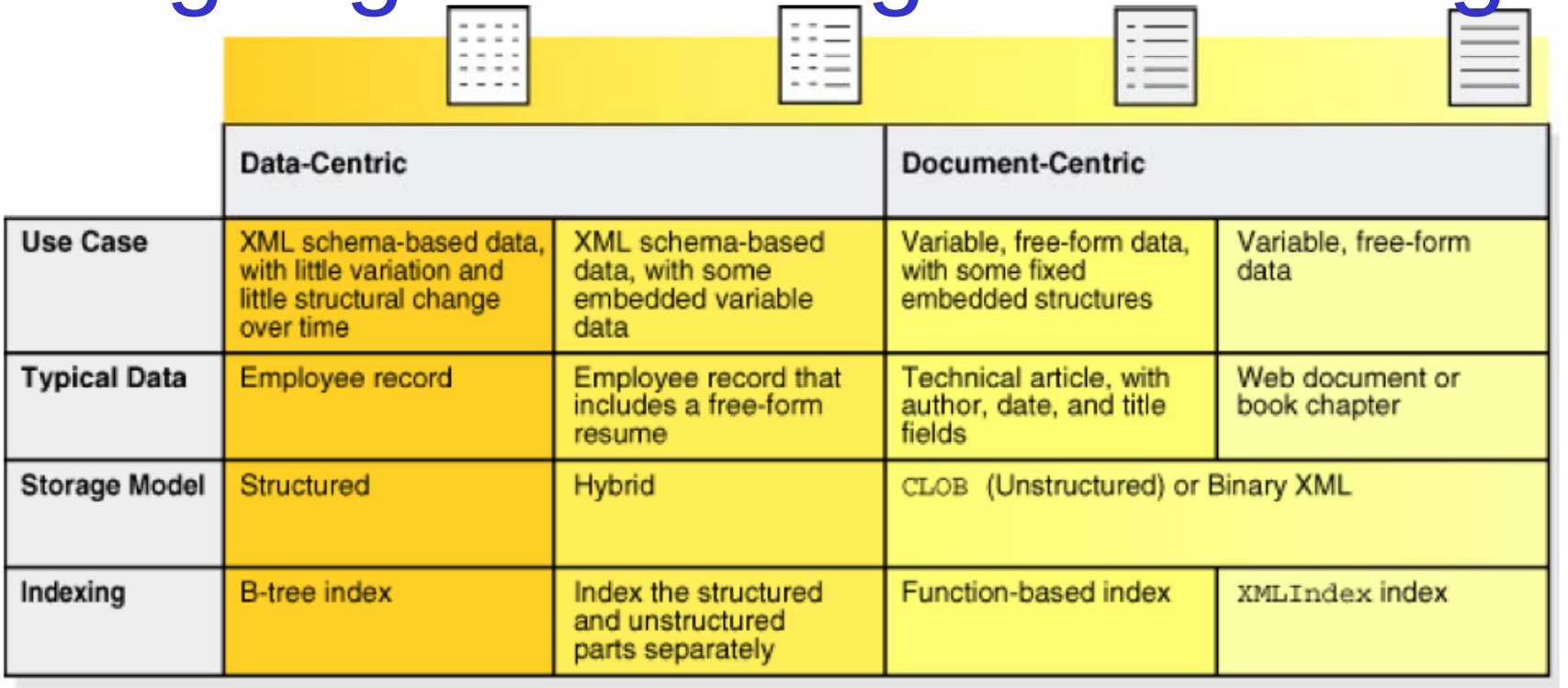
Eksempel på DBMS og håndtering av XML: Oracle XML DB

- Lagring av XML-formaterte dokumenter og data
  - Datasentriske eller dokumentsentriske XML-data
- ETL (Extract, Transform and Load)
  - Persistering av XML før transformering
- Eksport av relasjonelle data
  - Generere XML fra relasjonelle data
  - Resulterende XML-data blir ikke persistert

Oracle XML DB: XMLType

- Abstrakt datatype: XMLType
  - For lagring/persistering av XML-data
- Lagringsmodell: Velges etter bruksmønster
  - Structured storage (objekt-relasjonell lagring)
  - Binary XML storage (kompakt, skjemafleksibilitet)
  - Unstructured storage (CLOB, bevarer XML-dokumentet uendret)
  - Hybrid storage (forskjellige deler av et XML-dokument har forskjellig lagringsmodell)
- Indeksering: Velges etter bruksmønster
  - B-trær, XMLIndex

Oracle XML DB: Valg av lagringsmodell og indeksering



	Data-Centric		Document-Centric	
Use Case	XML schema-based data, with little variation and little structural change over time	XML schema-based data, with some embedded variable data	Variable, free-form data, with some fixed embedded structures	Variable, free-form data
Typical Data	Employee record	Employee record that includes a free-form resume	Technical article, with author, date, and title fields	Web document or book chapter
Storage Model	Structured	Hybrid	`CLOB` (Unstructured) or Binary XML	
Indexing	B-tree index	Index the structured and unstructured parts separately	Function-based index	`XMLIndex` index

Fra Oracle White Paper (Dec. 2009):
”Oracle XML DB: Choosing the Best XMLType Storage Option for Your Use Case”