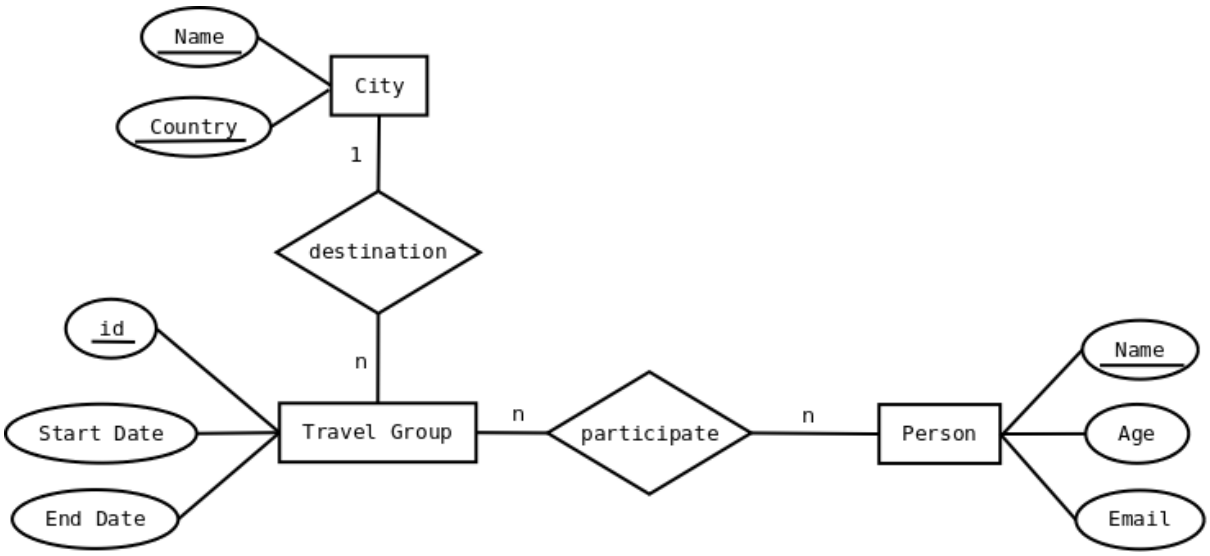
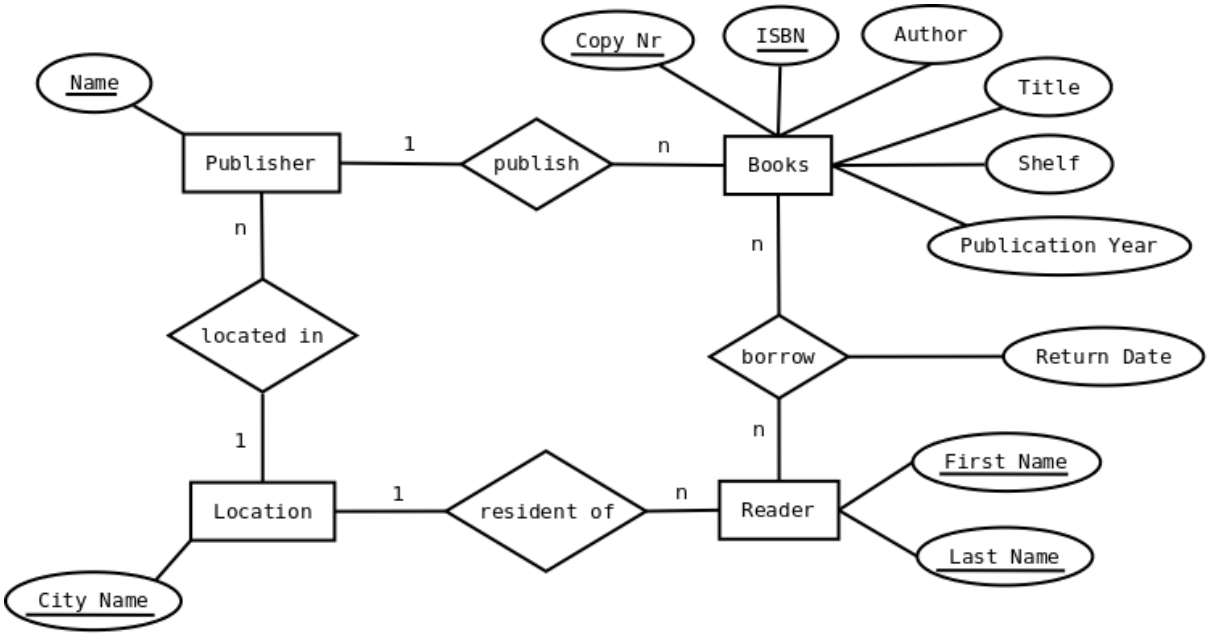
1 ER Modeling

1.1 People and Traveling



1.2 Library



2 Relational Model

2.1 ER to Relational Model

CITY (<u>CityName</u>, <u>CounrtyName</u>)
TRAVELGROUP (<u>GroupId</u>, CityName, CounrtyName, StartDate, EndDate)
PERSON (<u>PersonName</u>, Age, Email)
PARTICIPATE (<u>GroupId</u>, <u>PersonName</u>)

LOCATION (<u>CityName</u>)
PUBLISHER (<u>PublisherName</u>, CityName)
READER (<u>FirstName</u>, <u>LastName</u>, CityName)
BOOKS (<u>ISBN</u>, <u>CopyNr</u>, ShelfLocation, PublicationYear, PublisherName, Title, AuthorName)
BORROW (<u>ISBN</u>, <u>CopyNr</u>, <u>FirstName</u>, <u>LastName</u>, ReturnDate)

2.2 Relational Algebra

- $$\pi_{\text{LastName}}(\sigma_{\text{City}='\text{Zurich}'}(\text{READERS}))$$

- $$\pi_{\text{Title,Author}}(\sigma_{\text{BOOK.PublisherName}=\text{PUBLISHER.PublisherName}}(\text{BOOK} \times \sigma_{\text{City}='\text{Zurich}'}(\text{PUBLISHER})))$$

  or (equivalently)

  $$\pi_{\text{Title,Author}}(\text{BOOK} \bowtie \sigma_{\text{City}='\text{Zurich}'}(\text{PUBLISHER}))$$

- $$\pi_{\text{BOOK.Title,BOOK.Author}}(\sigma_{\text{BOOK.ISBN}=\text{BORROW.ISBN}}(\text{BOOK} \times \sigma_{\text{BORROW.ReaderNr}=\text{READER.RDNR}}(\text{BORROW} \times \sigma_{\text{FirstName}='\text{John}' \wedge \text{LastName}='\text{Doe}'}(\text{READER}))))$$

  or (equivalently)

  $$\pi_{\text{BOOK.Title,BOOK.Author}}(\text{BOOK} \bowtie \text{BORROW} \bowtie_{\text{BORROW.ReaderNr}=\text{READER.RDNR}} \sigma_{\text{FirstName}='\text{John}' \wedge \text{LastName}='\text{Doe}'}(\text{READER}))$$

4 SQL queries

1.
```
SELECT avg(s.Semester)
FROM Studenten s
WHERE 1
```

2.
```
SELECT *
FROM Studenten s
WHERE s.Semester > (
SELECT avg(s.Semester)
FROM Studenten s
WHERE 1)
```

3.
```
(SELECT PersNr as Id, Name FROM Professoren)
UNION
(SELECT PersNr as Id, Name FROM Assistenten)
UNION
(SELECT Legi as Id, Name FROM Studenten)
ORDER BY Id DESC
```

4.
```
SELECT DISTINCT s.Name
FROM Studenten s
JOIN hören h ON s.Legi = h.Legi
JOIN Vorlesungen v ON h.VorlNr = v.VorlNr
JOIN Professoren p ON v.gelesenVon = p.PersNr
WHERE p.Name = 'Sokrates'
```

5.
```
SELECT v.gelesenVon, p.Name, sum(v.KP)
FROM Vorlesungen v, Professoren p
WHERE v.gelesenVon = p.PersNr
GROUP BY v.gelesenVon, p.Name
```

6.
```
SELECT v2.Titel
FROM Vorlesungen v1
JOIN voraussetzen vor ON v1.VorlNr = vor.Vorgänger
JOIN Vorlesungen v2 ON vor.Nachfolger = v2.VorlNr
WHERE v1.Titel = 'Grundzuege'
```

Bonus: No SQL cannot handle transitivity. However, a lot of vendors offer custome extentions. Here the Oracle CONNECT BY clause:

```
SELECT Titel
FROM Vorlesungen
WHERE VorlNr in (
SELECT VorgÃ¤nger
FROM voraussetzen
CONNECTED BY Nachfolger = PRIOR Vorgänger
START WITH Nachfolger = (
SELECT VorlNr
from Vorlesungen
where Titel = '...' ));
```

7.
```
SELECT s.Legi, s.Name, count(*) as NumBekannter
FROM Studenten s, (
SELECT DISTINCT h1.Legi as Student, h2.Legi as Bekannter
```

```
FROM hören h1, hören h2
WHERE h1.VorlNr = h2.VorlNr AND h2.Legi <> h1.Legi
) b
WHERE s.Legi = b.Student
GROUP BY s.Legi, s.Name
ORDER BY NumBekannter DESC;
```