Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

# Informatics 2

## Assignment 12

# This week's exercises *(not to be fully shown in class before the end of assignment 12)*

# Exercise 12.1: JDBC – Database Access in Java

## Installation of the JDBC driver

- Create a new Java Project in Eclipse
- Create a new subdirectory „/lib“
- Download the database connector from the course website and store it in the „/lib“ folder
- Refresh your project (F5)
- Import DB-connector into your project:
  „Properties“ / „Java Build Path“ / „Add JARs“

# Exercise 12.1: JDBC – Database Access in Java

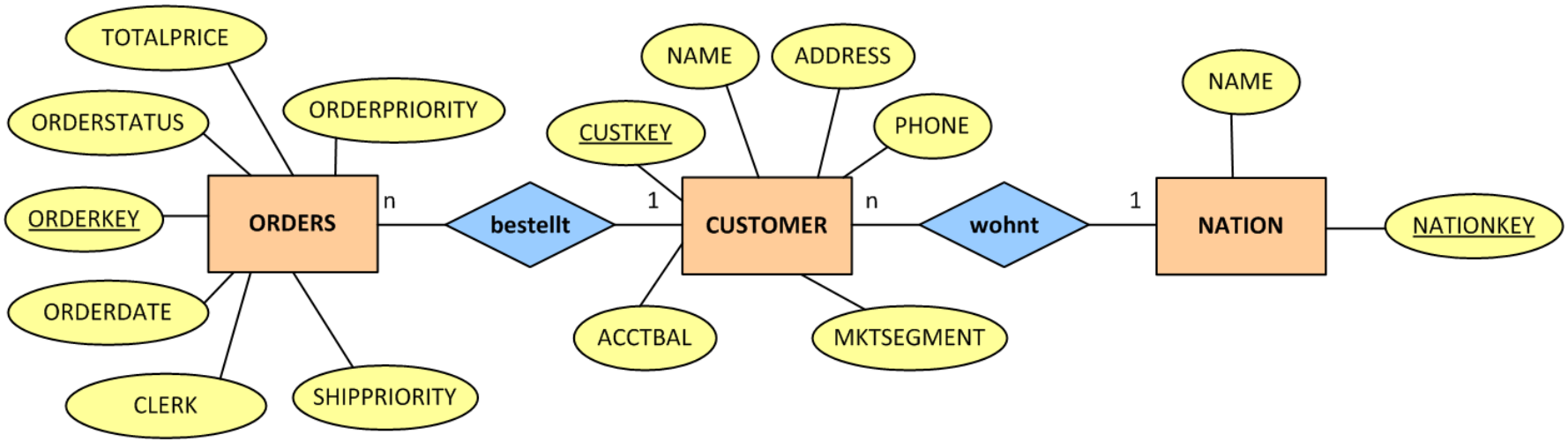
## A simple example:

```
// Connect to database server
Class.forName("com.mysql.jdbc.Driver");
Connection conn = DriverManager.getConnection(DB_URL,
  NETZ_USER, DB_PASS);

// Execute a query
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery("SELECT COUNT(*) AS num FROM
customer");
while( rs.next() ) {
       System.out.println("result: " + rs.getInt("num") );
}

// Close the result set, statement and the connection
rs.close();
stmt.close();
```

# Exercise 12.1: JDBC – Database Access in Java

## Example query

- **Schema:**

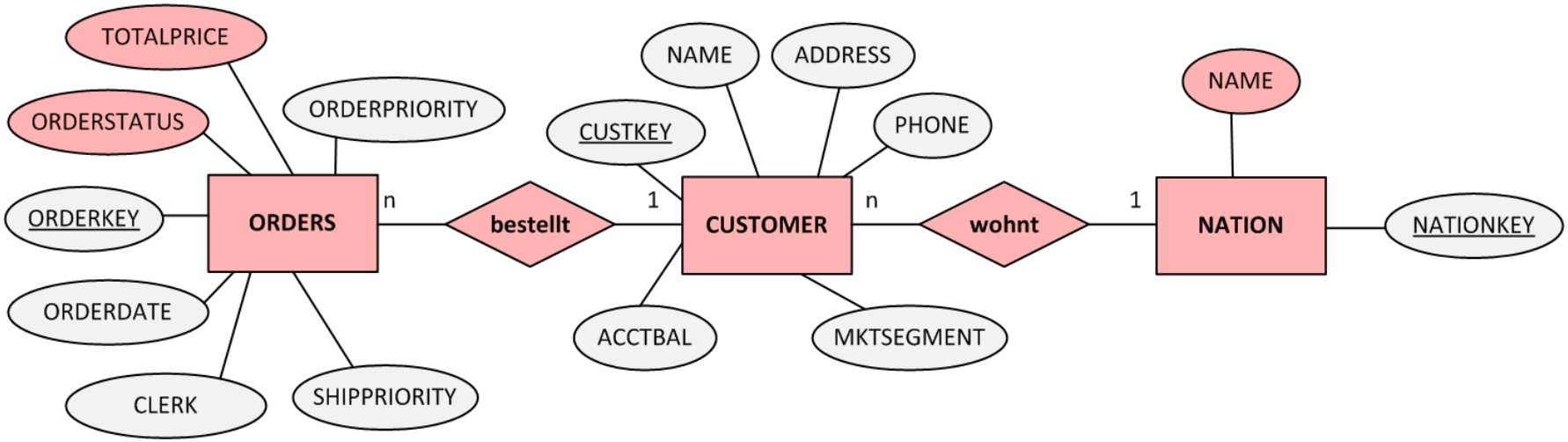



- „What‘s the name of the country, which has the highest value of undelivered orders?“

# Exercise 12.1: JDBC – Database Access in Java

## Example query

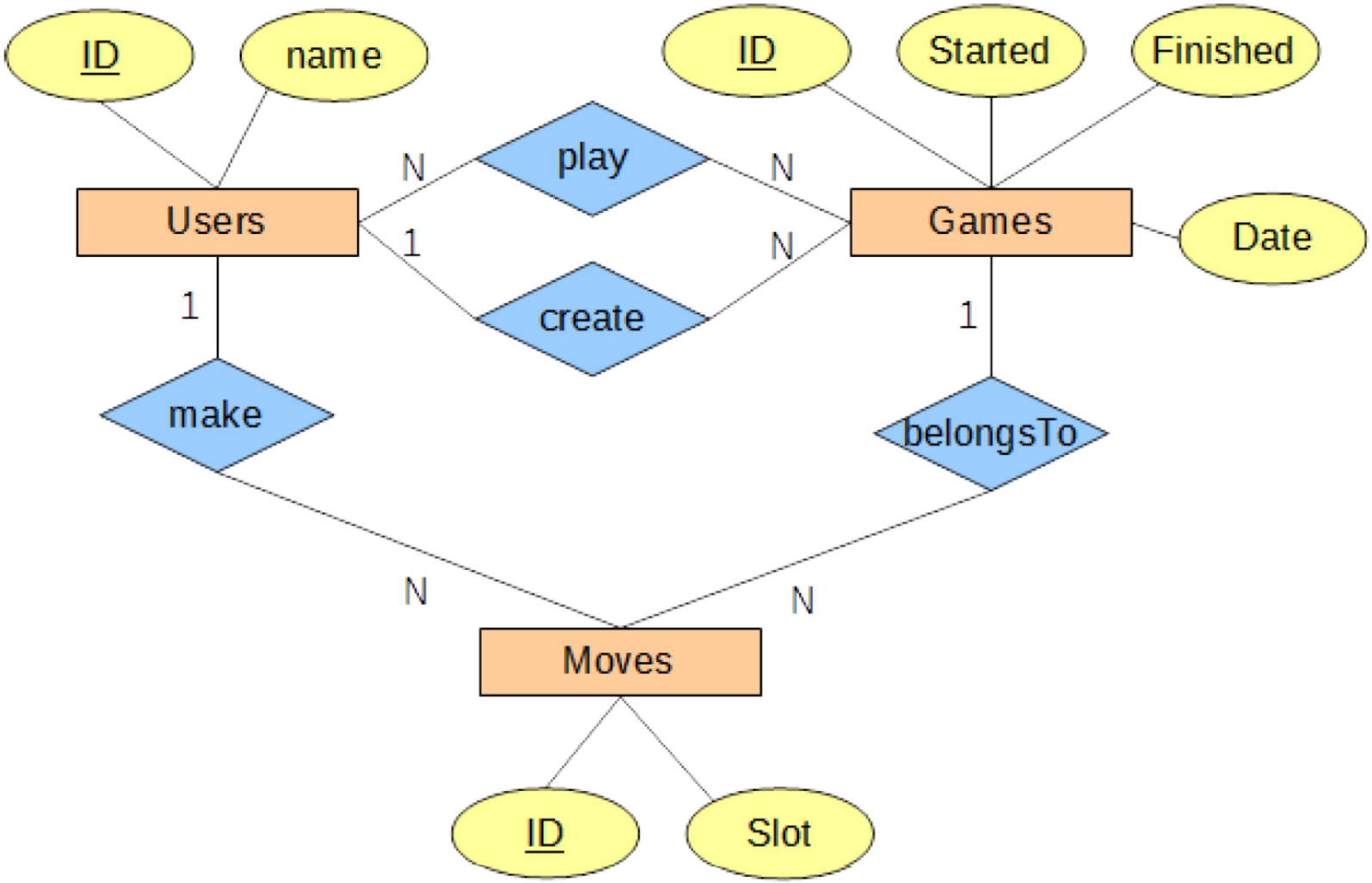
- **For the question this part of the scheme is relevant:**




- „What‘s the name of the country, which has the highest value of undelivered orders?“

# Exercise 12.2: Connect Four (Four in a Row)

## Schema



# Exercise 12.2 (a): Relations

## Entities

- **Users** (<u>userID</u>, name)
- **Games** (<u>gameID</u>, started, finished, date)
- **Moves** (<u>moveID</u>, slot)

# Exercise 12.2 (a): Relations

## Relationships

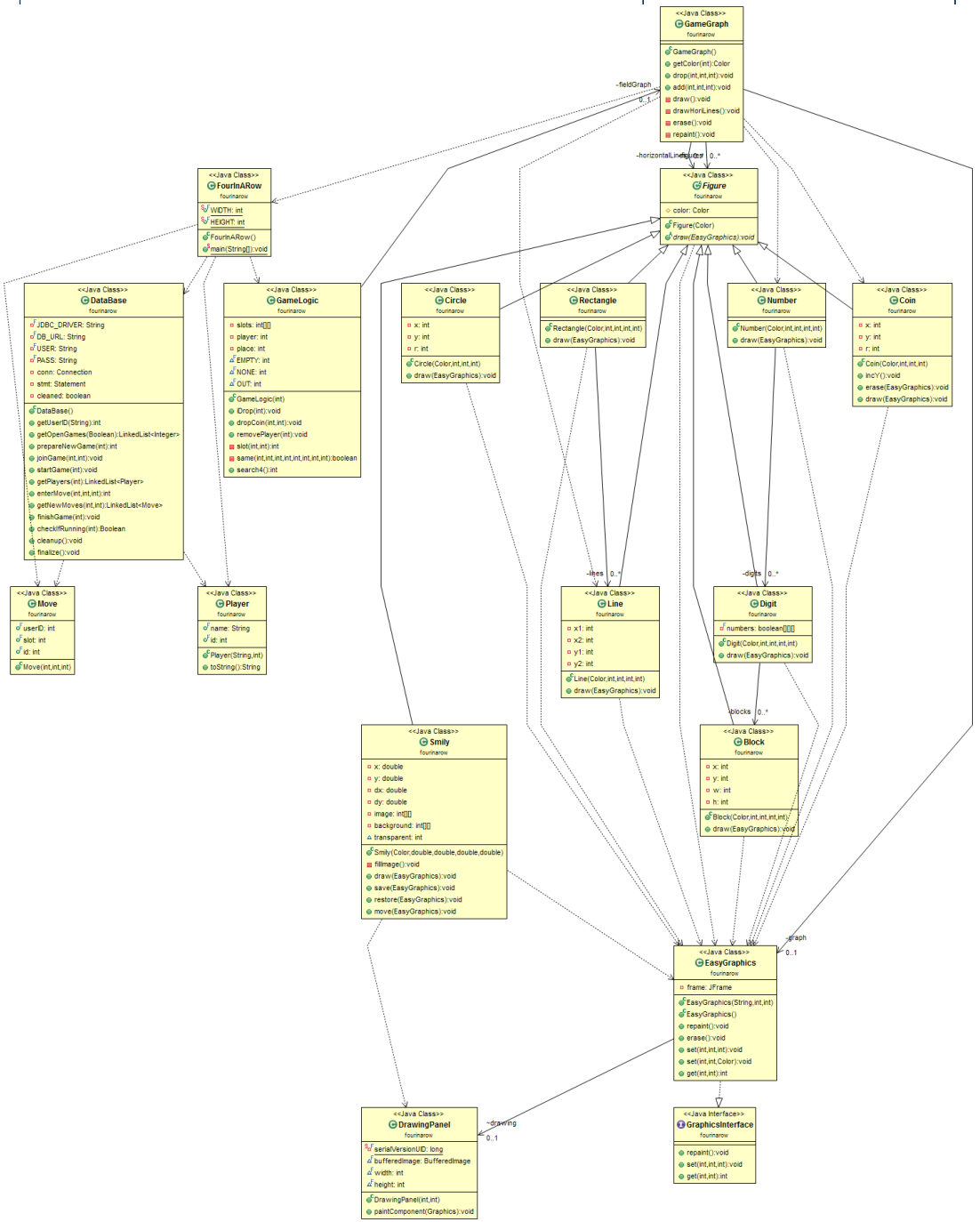
- **play** (gameID, userID)
- **create** (gameID, createdByUserID)
- **make** (moveID, userID)
- **belongsTo** (moveID, gameID)

# Exercise 12.2 (a): Relations

## Combine relations with the same keys

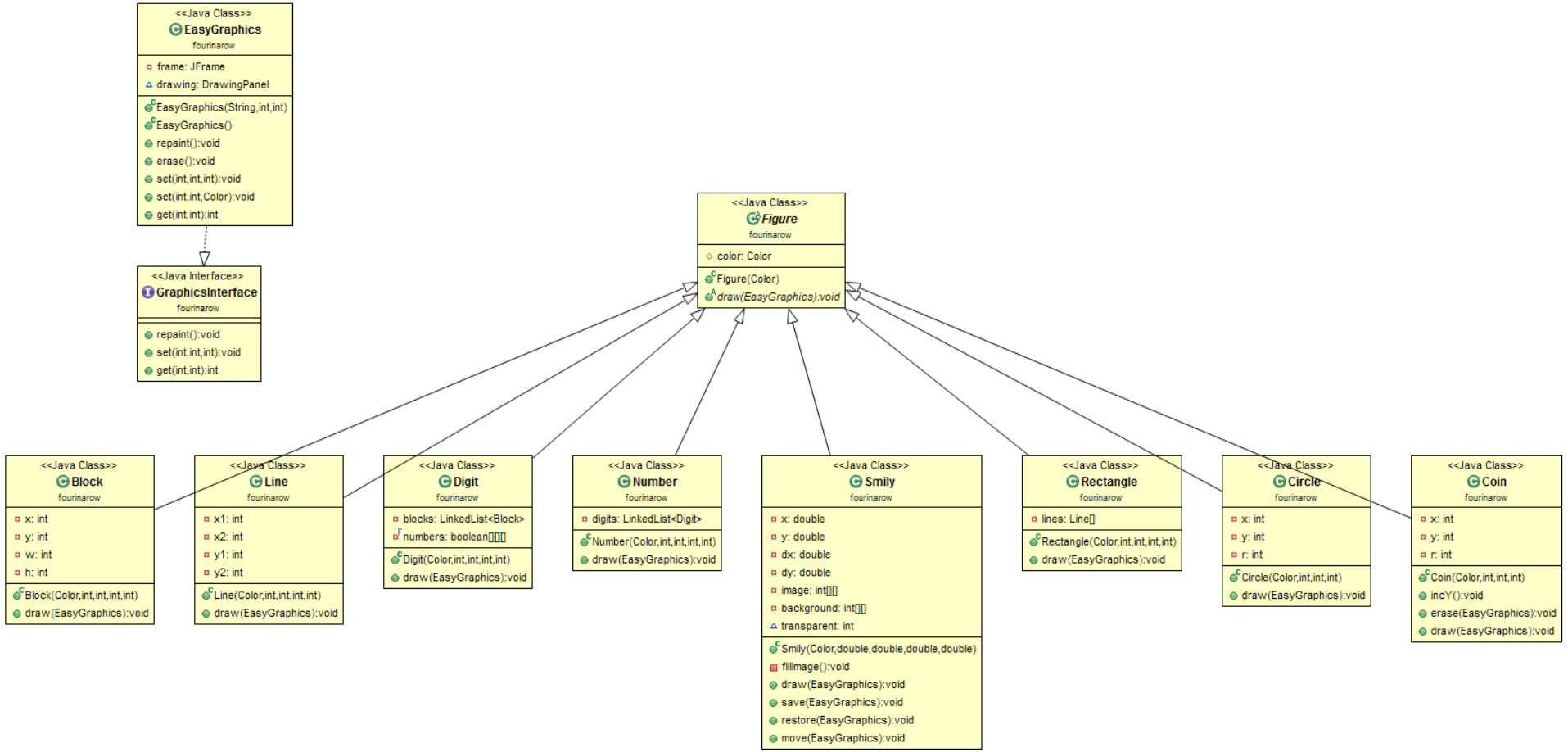
- **Users** (<u>userID</u>, name)
- **Games** (<u>gameID</u>, started, finished, date, createdByUserID)
- **Moves** (<u>moveID</u>, slot, userID, gameID)
- **play** (<u>gameID</u>, <u>userID</u>)

# Exercise 12.2 (c): Class Diagram




Tool: ObjectAid UML Explorer

# Exercise 12.2 (c): Class Diagram (Inheritance)



# Exercise 12.2 (d): Queries

## getUserID()

### Query: Lines from table Users where Name=<name>

```
"SELECT * FROM Users
WHERE Name = \""+name+"\""
```

### Insert: New line (Name, GameID) in table Users with values (<name>, <gameId>)

```
"INSERT INTO Users (Name, GameID)
VALUES
(\""+name+"\", \""+gameID+"\")"
```

# Exercise 12.2 (d): Queries

## Query: Lines of table Games with ID = <gameID>

```
"SELECT * FROM Games
WHERE ID = \""+gameID+"\""
```

## getOpenGames()

## Query: Column ID of the lines from table Games where Started = 0

```
"SELECT ID FROM Games WHERE Started = 0"
```

# Exercise 12.2 (d): Queries

## prepareNewGame()

**Insert: New line (Started, Finished, Creater) in table Games with the values (0, 0, <userID>)**

```
"INSERT INTO Games (Started, Finished,
Creator) VALUES (0, 0, \""+userID+"\")"
```

**Query: Column ID of the lines from table Games where Started = 0, Finished = 0 und Creator = <userID>, ordered by ID**

```
"SELECT ID FROM Games WHERE Started = 0
  AND Finished = 0
  AND Creator = \""+userID+"\" ORDER BY ID"
```

# Exercise 12.2 (d): Queries

## joinGame()

**Update: Column GameID gets the new value <gameID> in table Users on the lines with ID = <userID>**

```
"UPDATE Users SET GameID = "+gameID+"
WHERE ID = "+userID
```

## startGame()

**Update: Column Started gets the new value 1 in table Games on the lines where ID = <gameID>**

```
"UPDATE Games SET Started = 1
WHERE ID = "+gameID
```

# Exercise 12.2 (d): Queries

## getPlayers()

**Query: Columns ID and Name of the lines from table Users where GameId = <gameID>, ordered by ID**

```
"SELECT ID, Name FROM Users
WHERE GameID = \""+gameID+"\" ORDER BY ID"
```

## enterMove()

**Insert: (UserID, Slot, GameId) in table Moves with the values (<userID>, <slot>, <gameID>)**

```
"INSERT INTO Moves (UserID, Slot, GameID)
VALUES (\""+userID+"\", \""+slot+"\",
\""+gameID+"\")"
```

# Exercise 12.2 (d): Queries

**Query: Column ID from the lines of table Moves where UserId = <userID>, Slot = <slot> und GameID = <gameID>, ordered by ID**

```
"SELECT ID FROM Moves WHERE UserID =
\""+userID+"\" AND Slot = \""+slot+"\" AND
GameID = \""+gameID+"\" ORDER BY ID"
```

**getNewMoves()**

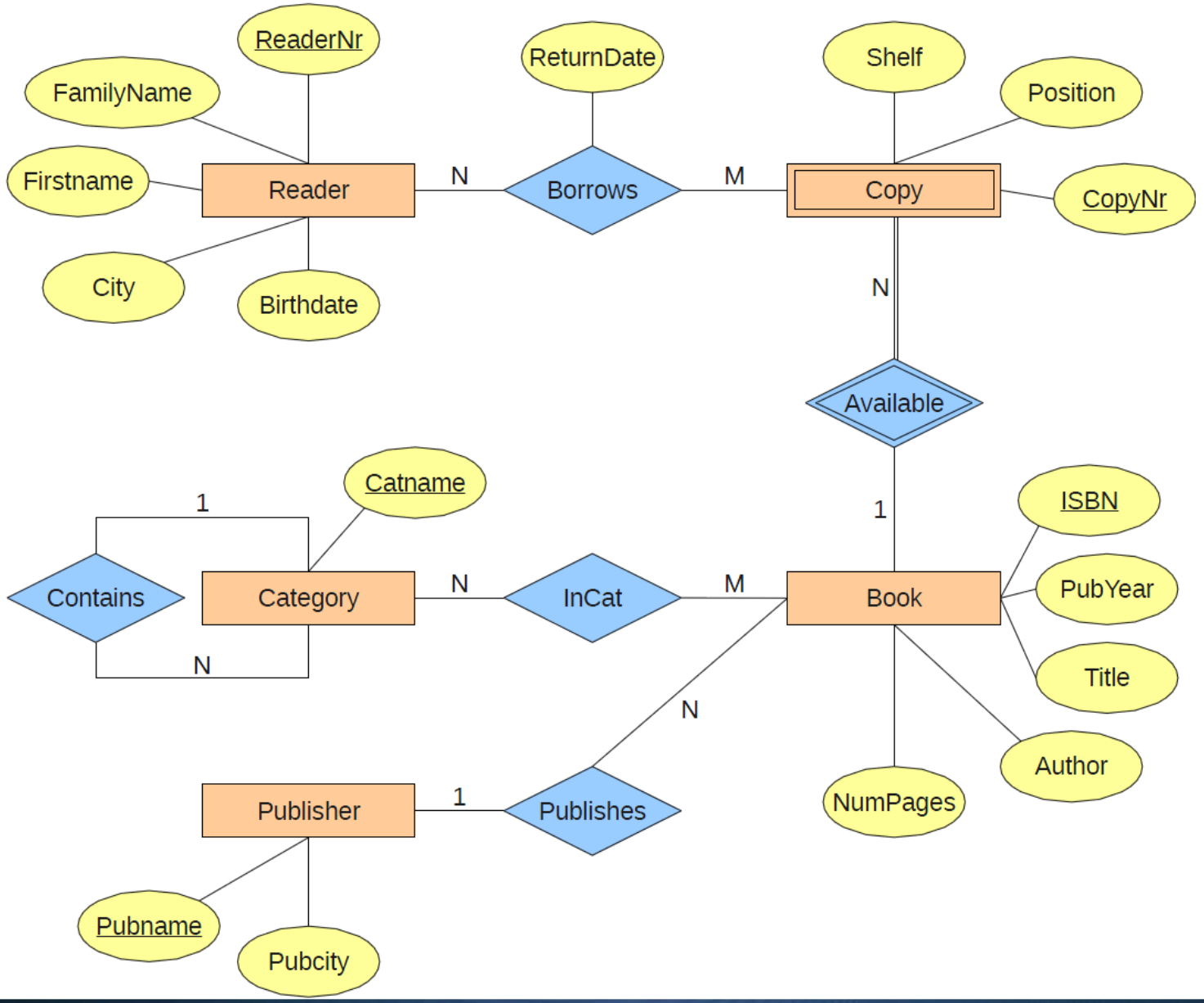
**Query: Columns UserID, Slot, ID from the lines of table Moves where GameID = <gameID> and ID = <lastID>, ordered by ID**

```
"SELECT UserID, Slot, ID FROM Moves
WHERE GameID = \""+gameID+"\" AND
ID > \""+lastID+"\" ORDER BY ID"
```

# Solution
# Assignment 11

# Exercise 11.1.1





# Exercise 11.1.1: Library

## Entities

- **Reader** (ReaderNr, FamilyName, Firstname, City, Birthdate)
- **Book** (ISBN,Title, Author, NumPages, PubYear)
- **Publisher** (Pubname, Pubcity)
- **Category** (Catname)
- **Copy** (ISBN, CopyNr, Shelf, Position)

# Exercise 11.1.1: Library

## Relationships

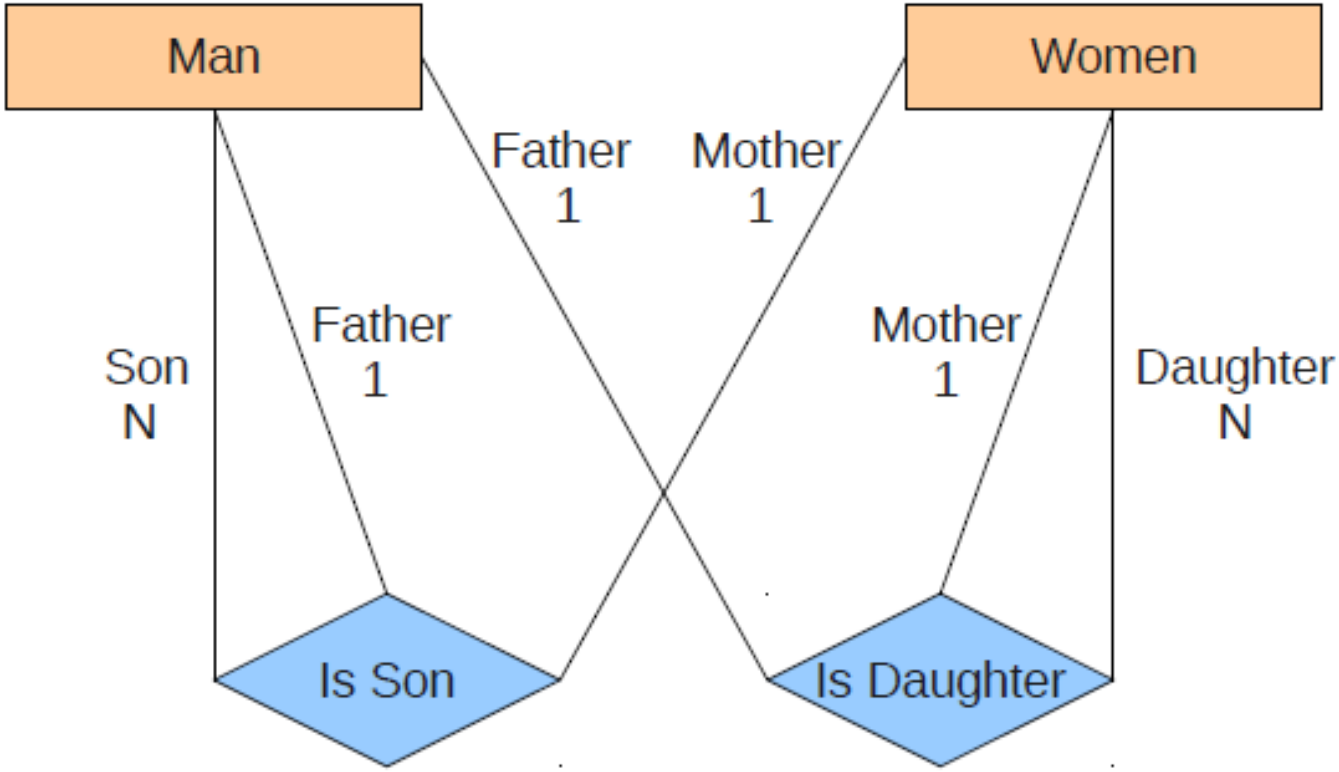
- **Borrows** (ReaderNr, ISBN, CopyNr, ReturnDate)
- **Available** (ISBN, CopyNr)
- **Contains** (Catname, ContainedIn)
- **InCat** (ISBN, Catname)
- **Publishes** (ISBN, Pubname)

# Exercise 11.1.1: Library

## Combine relations with the same key

- **Reader** (ReaderNr, FamilyName, Firstname, City, Birthdate)
- **Book** (ISBN, Title, Author, NumPages, PubYear, Pubname)
- **Publisher** (Pubname, Pubcity)
- **Category** (Catname, ContainedIn)
- **Copy** (ISBN, CopyNr, Shelf, Position)
- **Borrows** (ReaderNr, ISBN, CopyNr, ReturnDate)
- **InCat** (ISBN, Catname)

# Exercise 11.1.2: Inheritance



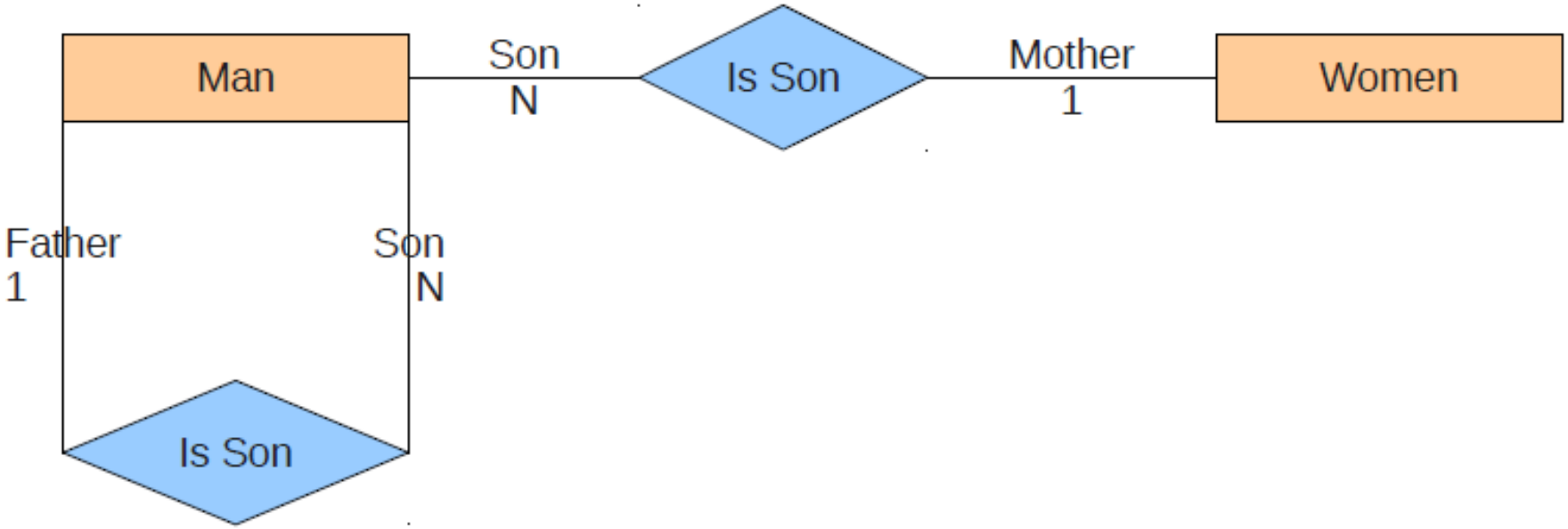
# Exercise 11.2: Inheritance – 1st way

- **Men** (MName)
- **Women** (WName)
- **Is_Son** (Son, Father, Mother)

  *or* **Is_Son** (Son, Father, Mother)
- **Is_Daughter** (Daughter, Father, Mother)

  *or* **Is_Daughter** (Daughter, Father, Mother)

# Exercise 11.2: Inheritance

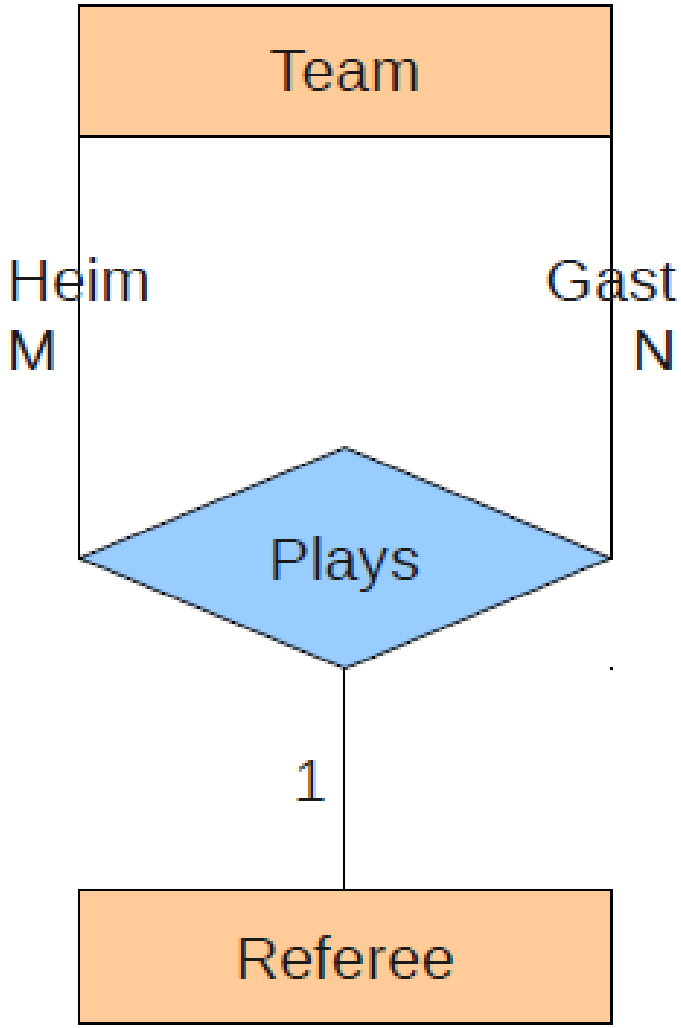



Analog for „**is Daughter**“

# Exercise 11.1.2: Inheritance – 2nd way

- **Men** (MName, Father, Mother)
- **Women** (WName, Father, Mother)

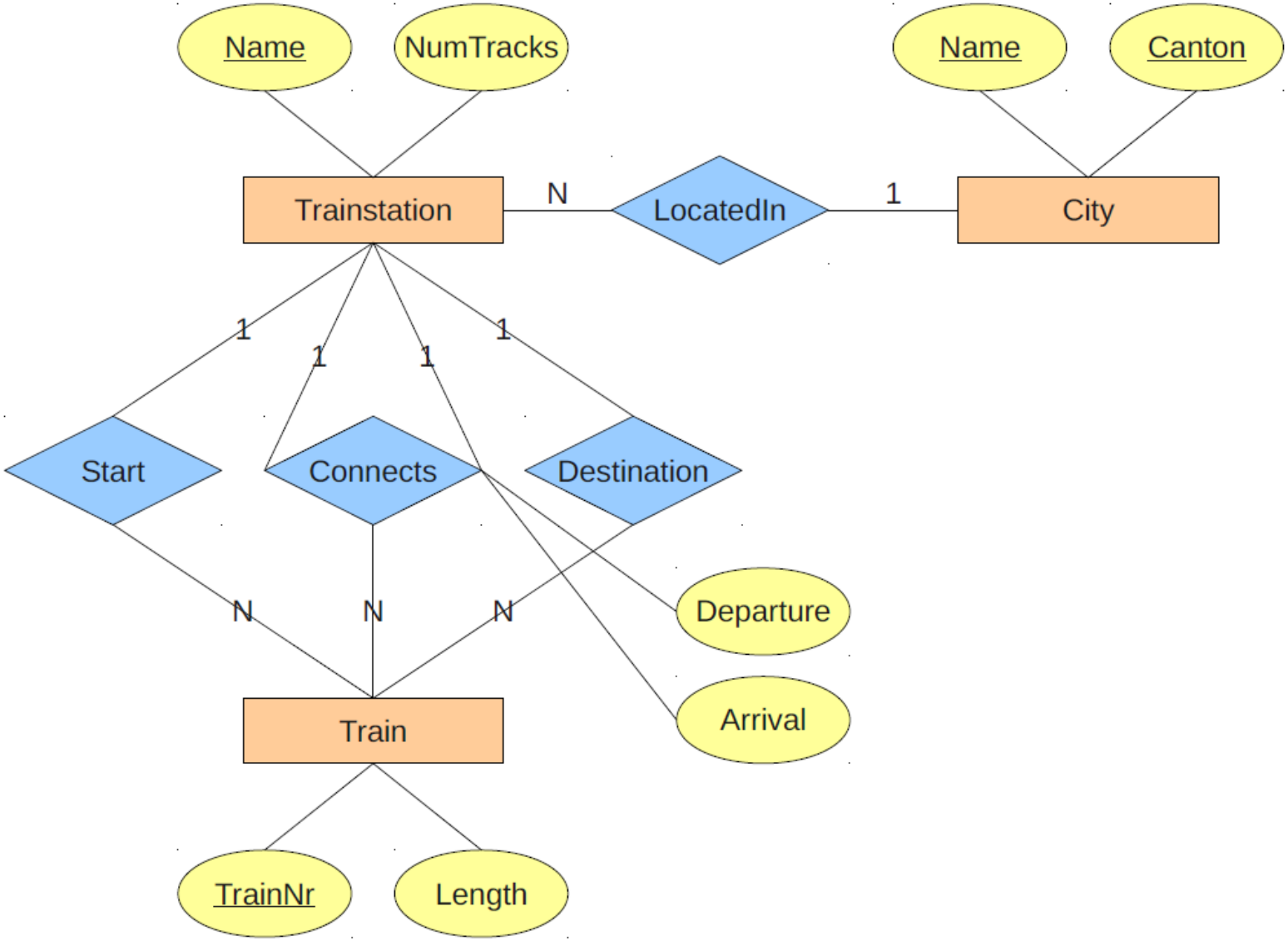
# Exercise 11.1.3: Football



# Exercise 11.1.3: Football

- **Team**: (Teamname)
- **Refree** (RName)
- **Play** (PName, Local, Guest)

# Exercise 11.1.4: Trains



# Exercise 11.1.4: Trains

## Entities

- **City** (Name, Canton)
- **Trainstation** (Name, NumTracks)
- **Train** (TrainNr, Length)

# Exercise 11.1.4: Trains

## Relationships

- **LocatedIn** (TrainstationName, CityName, Canton)
- **Start** (TrainNr, StartTrainstationName)
- **Destination** (TrainNr, DestTrainstationName)
- **Connects** (FromTrainstation, ToTrainstation, TrainNr, Departure, Arrival)

  or

- **Connects** (FromTrainstation, ToTrainstation, TrainNr, Departure, Arrival)

# Exercise 11.1.4: Trains

## Combine relations with the same key

- **City** (Name, Canton)
- **Trainstation** (Name, NumTracks, CityName, Canton)
- **Train** (TrainNr, Length, StartTrainstationName, DestTrainstationName)
- **Connects** (FromTrainstation, ToTrainstation, TrainNr, Departure, Arrival)

  or

- **Connects** (FromTrainstation, ToTrainstation, TrainNr, Departure, Arrival)

# The relational Algebra

- $\sigma$ Selection
- $\pi$ Projection
- $\times$ cartesian product
- $\bowtie$ Join
- $\rho$ Renaming

# Exercise 11.2: Relational Algebra

## Schema

- **Reader** ( RDNR, Surname, Firstname, City, Birthdate )
- **Book** ( ISBN, Title, Author, NoPages, PubYear, PublisherName )
- **Publisher** ( PublisherName, PublisherCity )
- **Category** ( CategoryName, BelongsTo )
- **Copy** ( ISBN, CopyNumber, Shelf, Position )
- **Loan** ( ReaderNr, ISBN, Copy, ReturnDate )
- **BookCategory** ( ISBN, CategoryName )

# Exercise 11.2: Relational Algebra

## Queries

**a) Find the lastnames of the readers in Zürich?**

$$\Pi_{Surname}(\sigma_{City=Zurich}(READER))$$

# Exercise 11.2: Relational Algebra

## Queries

**b) Which books (Author, Title) stem from publishers in Zürich, Berne or New York?**

$$\Pi_{Author,Title}(BOOK \bowtie (\sigma_{City=Zurich OR City=Bern OR City=New York}(PUBLISHER)))$$

# Exercise 11.2: Relational Algebra

## Queries

**c) Which books (Author, Title) has reader Lemmi Schmöker borrowed?**

$$\Pi_{Author,Title}(BOOK \bowtie LOAN \bowtie_{ReaderNr=RDNR} (\sigma_{Surname=Schmoker AND Firstname=Lemmi}(READER)))$$

# Exercise 11.2: Relational Algebra

## Queries

**d) Which books in the category „Alps“ don‘t belong to the category „Switzerland“?**

$$\pi_{\text{ISBN}}(\sigma_{\text{CategoryName}\neq\text{Alps}}(\rho_{b1}(\text{BookCategory})))\text{A}$$
$$\pi_{\text{ISBN}}(\sigma_{\text{CategoryName}=\text{Switzerland}}(\rho_{b2}(\text{BookCategory})))$$

# Exercise 11.2: Relational Algebra

## Queries

**e) Which readers (Surname, Firstname) have borrowed books, which have been published in the same place where they live?**

$$\Pi_{Firstname,Surname}(\sigma_{City=PublisherCity}(PUBLISHER \bowtie BOOK \bowtie LOAN \bowtie_{ReaderNr=RDNR} READER))$$

# Exercise 11.2: Relational Algebra

## Queries

**f) Which readers have borrowed at least one book, which has been lent to Lemmi Schmöker as well?**

$$\pi_{\text{R1.Firstname,R1.Surname}} ($$
$$(\rho_{\text{R1}}(\text{READER}) \; {}^{\text{A}}_{\text{ReaderNr=RDNR}} \; \rho_{\text{L1}}(\text{LOAN}))$$
$${}^{\text{A}}_{\text{R1.RDNR}\neq\text{R2.RDNR,L1.ISBN=L2.ISBN}}$$
$$(\rho_{\text{R2}}(\sigma_{\text{Surname=Schmoker and Firstname=Lemmi}}(\text{READER}))$$
$${}^{\text{A}}_{\text{ReaderNr=RDNR}} \; \rho_{L2}(\text{LOAN})$$
$$)$$

# Exercise 11.3: SQL Queries

## Queries

**a) Which are the lastnames of the readers in Zürich?**

```
SELECT DISTINCT Surname
FROM Reader
WHERE City = 'Zurich'
ORDER BY Surname DESC
```

# Exercise 11.3: SQL Queries

## Queries

**b) Which books (Author, Title) stem from publishers in Zürich, Bern or New York?**

```
SELECT Author, Title
FROM Book B, Publisher P
WHERE B.PublisherName =
    P.PublisherName
  AND (P.PublisherCity = 'Zurich'
  OR P.PublisherCity = 'Bern'
  OR P.PublisherCity = 'New York')
```

# Exercise 11.3: SQL Queries

## Queries

**c) Which books (Author, Title) has reader Lemmi Schmöker borrowed?**

```
SELECT B.Author, B.Title
FROM Reader R, Loan L, Book B
WHERE R.Surname = 'Schmöker'
  AND R.Firstname = 'Lemmi'
  AND R.RDNR = L.ReaderNr
  AND L.ISBN = B.ISBN
```

# Exercise 11.3: SQL Queries

## Queries

**d) Which books in the categorie "Alps" don‘t belong to the categorie "Switzerland"?**

```
select ISBN
from BookCategory b1, BookCategory b2
where b1.CategoryName != 'Alps'
and b2.CategoryName = 'Switzerland'
and b1.ISBN = b2.ISBN
```

# Exercise 11.3: SQL Queries

## Queries

**e) Which readers (Surname, Firstname) have borrowed books, which have been published in the same place where they live?**

```
SELECT R.Firstname, R.Surname
FROM Reader R, Loans L, Book B,
   Publisher P
WHERE R.RDNR = L.ReaderNr
   AND L.ISBN = B.ISBN
   AND B.PublisherName =
      P.PublisherName
   AND R.City = P.PublisherCity
```

# Exercise 11.3: SQL Queries

## Queries

**f) Which readers have borrowed at least one book, which has been lent to Lemmi Schmöker as well?**

```
SELECT R1.Firstname, R1.Surname
FROM Reader R1, Loan L1, Loan L2,
   Reader R2
WHERE R2.Firstname='Lemmi'
   AND R2.NAME = 'Schmöker'
   AND L2.ReaderNr = R2.RDNR
   AND R1.RDNR != R2.RDNR
   AND R1.ReaderNr = L1.RDNR
```

# Exercise 11.4: SQL Updates

## Queries

**a) Insert a new Nation with the Name „Switzerland“.**

```
INSERT INTO nation
(name) VALUES ('Switzerland')
```

# Exercise 11.4: SQL Updates

## Queries

**b) Delete all orders with a total value (totalprice) with less than 100.**

```
DELETE FROM orders
WHERE totalprice < 100
```

# Exercise 11.4: SQL Updates

## Queries

**c) Alter the status of an order (orderstatus) with number (orderkey) 4 from "O" to "F".**

```
UPDATE orders
SET orderstatus = 'F'
WHERE orderkey = 4
```