

# Database Management System (DBMS)

**L-6:**

**Entity Relationship Modelling**

*Dr. Kamruddin Nur*

*Associate Professor, Computer Science*

*kamruddin.nur@gmail.com*

# ***Lecture Content***

- Entity Relationship (ER) Model
- Entities, Attributes, Relationships
- Degree of Relationships
- Single-value, Multi-value attributes
- Strong, Weak entities
- Keys
- Structural Constraints

# ***Why Modelling ?***

- ◆ *The fact that database designers, programmers, and end-users view the data and its use differently*
- ◆ *To solve this problem we need to have a model for communication which is -*
  - *non-technical*
  - *free of ambiguities*
  - *Entity-Relationship (ER) model is one of them*

# ***ER Model***

*Top-down approach following the steps -*

- ➔ Identify important data called **entities***
- ➔ Identify **relationships** between data*
- ➔ Identify **attributes** of entities*
- ➔ Identify **constraints** on entities, relationships and attributes*

# ***Entity Types***

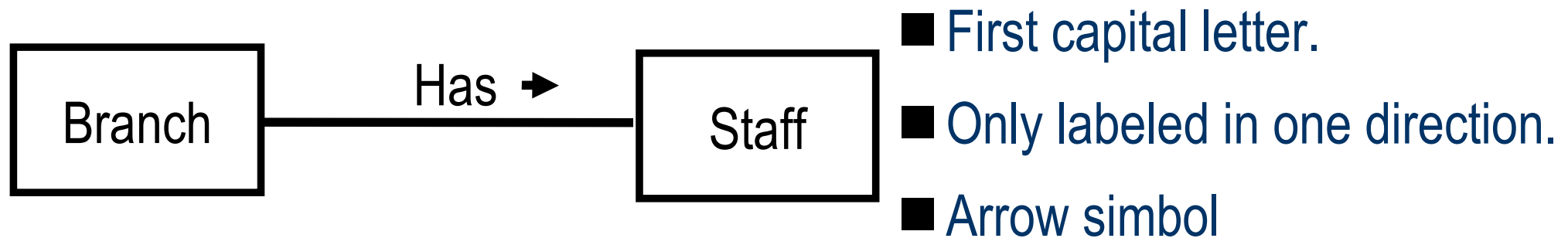
- **Entity Type**: A group of objects with the same properties, which are identified by the enterprises as having an independent existence.
- **Entity Occurrence**: A uniquely identifiable object of a entity type.



- Rectangle labeled with the name of the entity.
- In UML representation, the first letter of the entity name is a capital letter.

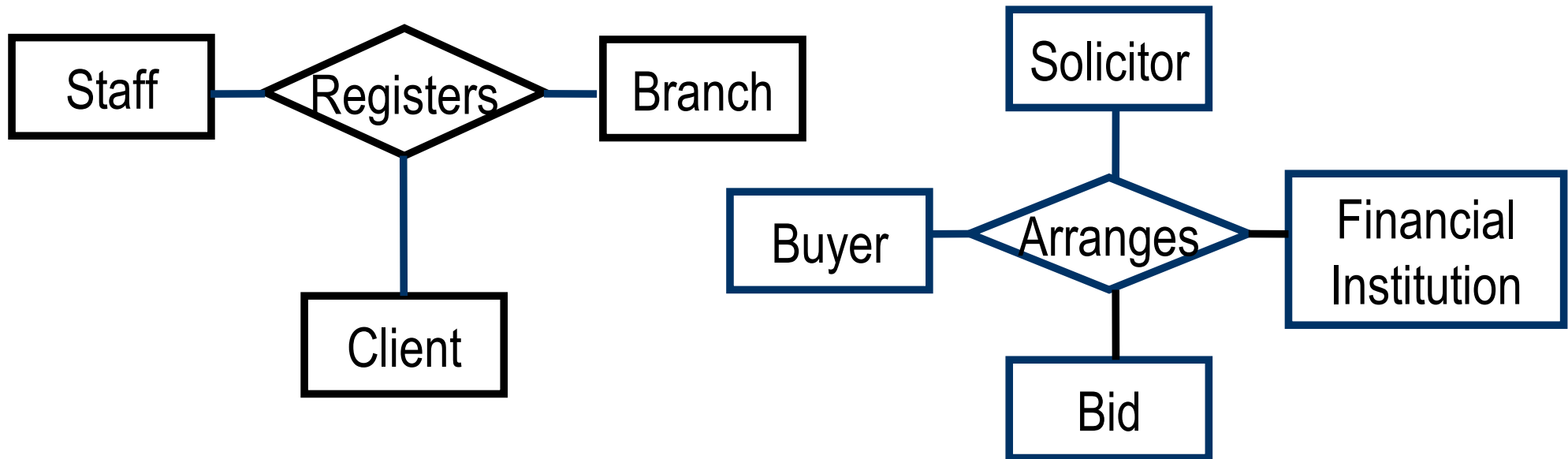
# Relationship Types

- **Relationship Type**: A set of meaningful associations among entity types.
- **Relationship Occurrence**: A uniquely identifiable association, which includes one occurrence from each participating entity type.



# ***Degree of Relationship Types***

- **Degree of Relationship type**: the number of participating entity types in a relationship.
- A relationship of degree two is called **binary**, a relationship of degree three is called **ternary**...

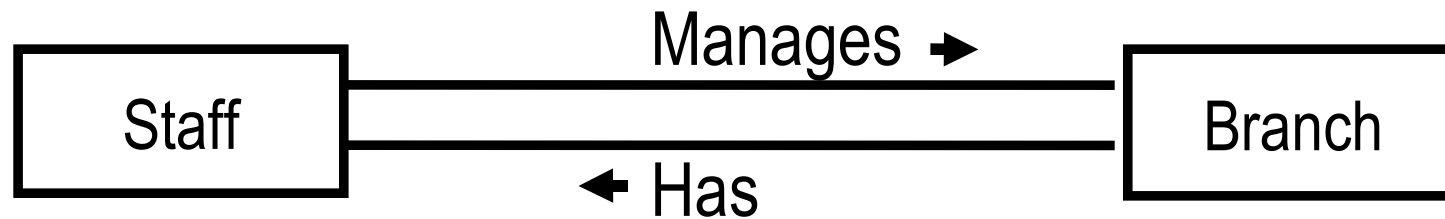


“Staff registers a client at a branch”

“A solicitor arranges a bid on behalf of a buyer supported by a financial institution”

# ***Recursive Relationship***

- **Recursive Relationship**: A relationship type where the same entity type participates more than once in different roles.





# ***Attributes***

- **Attribute**: A property of an entity or a relationship type. For example: staffNo, name, position... To describe the entity Staff.
- **Attribute Domain**: The set of allowable values for one or more attributes.
- Attributes can be classified as being: *simple* or *composite*; *single-valued* or *multi-valued*; or *derived*.

# ***Simple and Composite Attributes***

- **Simple Attribute:** An attribute composed of a single component with an independent existence. E.g position and salary of the Staff entity.
- **Composite Attribute:** An attribute composed of multiple components, each with an independent existence.
- E.g adress attribute of the branch entity that can be subdivided into street, city and postcode attributes.

# ***Single-Valued and Multi-Valued Attributes***

- **Single-Valued Attribute:** An attribute that holds a single value for each occurrence of an entity type. E.g branchNo.
- **Multi-Valued Attributes:** An attribute that holds multiple values for each occurrence of an entity type. E.g telephoneNo.

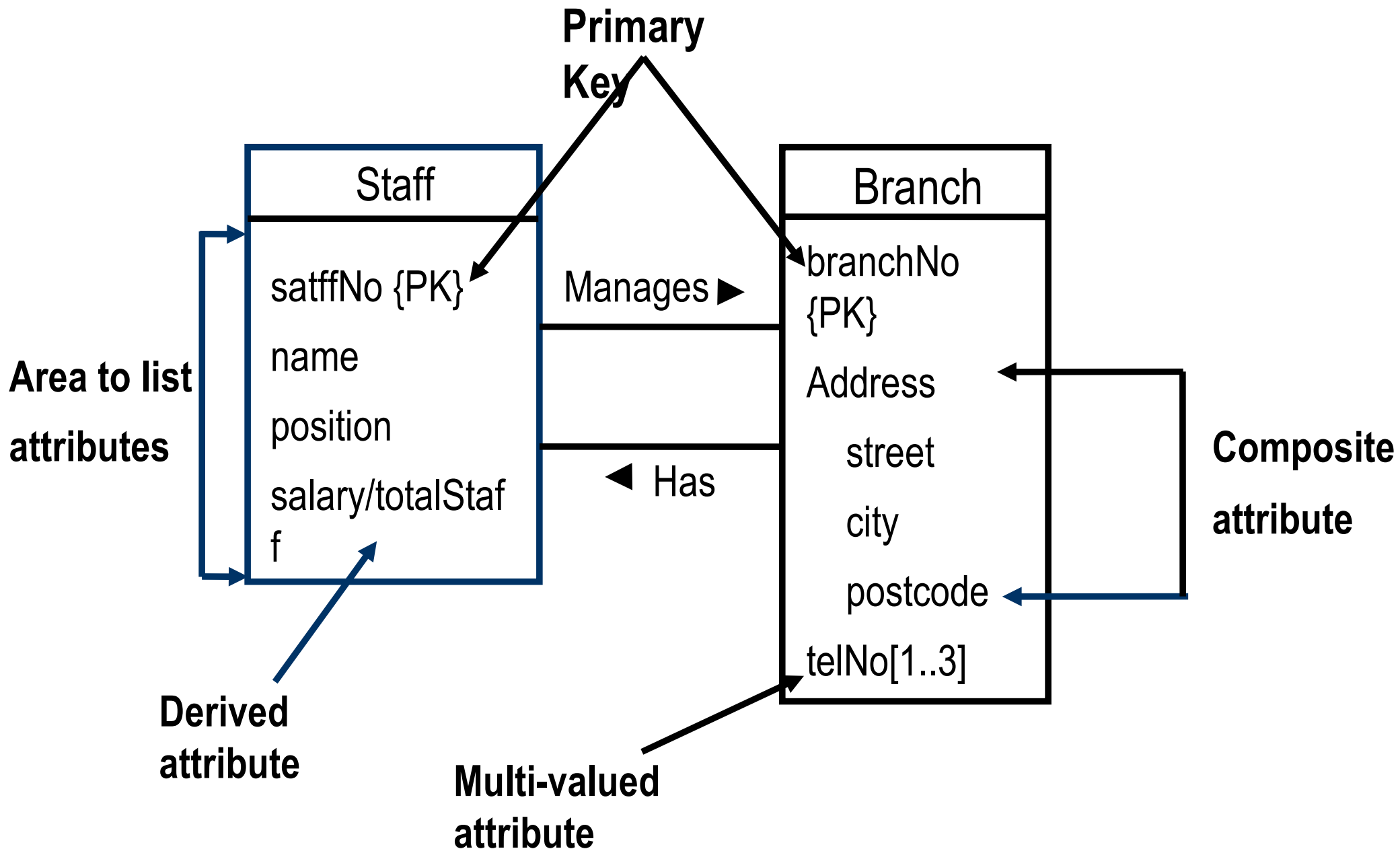
# ***Derived Attributes***

- **Derived Attributes:** An attribute that represents a value that is derivable from the value of a related attribute or set of attributes, not necessarily in the same entity type.
- E.g attribute duration which value is derived from the rentStart and rentFinish attributes.

# Keys

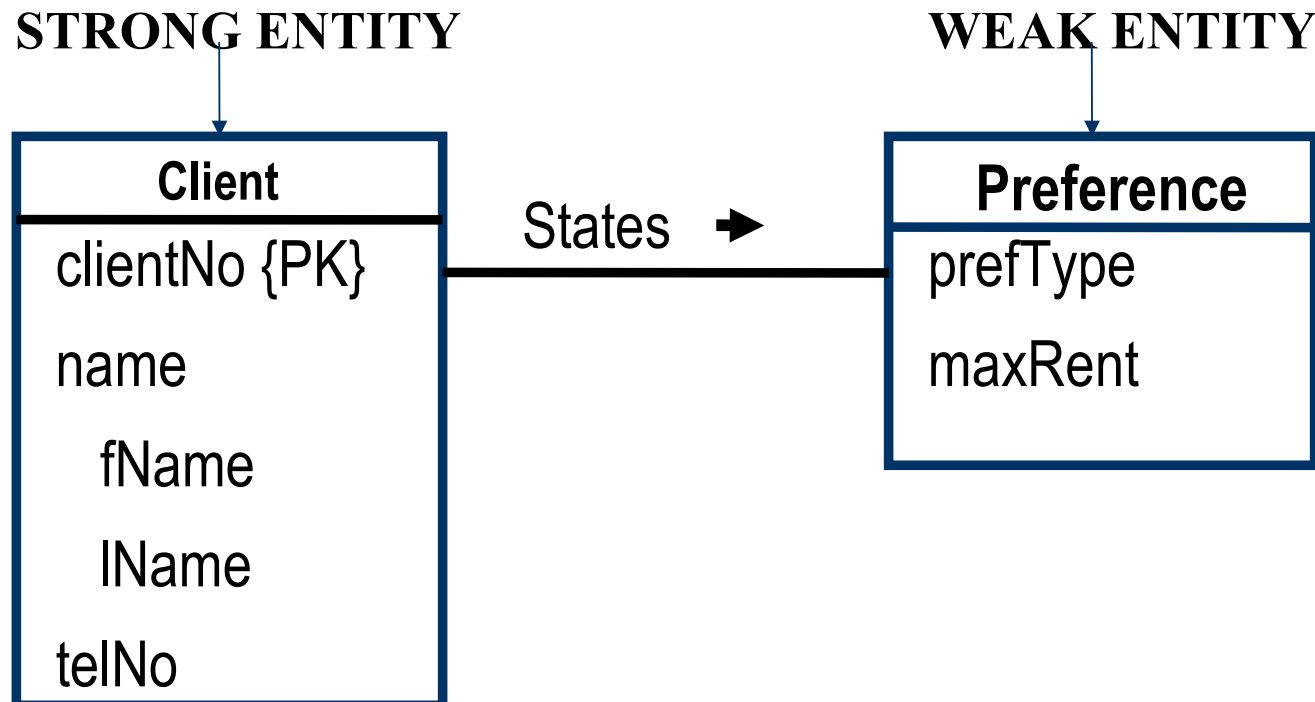
- **Candidate Key** (never NULL): The minimal *set* of attributes that uniquely identifies each occurrence of an entity type.  
E.g: branchNo in entity Branch.
- **Primary Key**: The candidate key that is selected to uniquely identify each occurrence of an entity type.  
E.g: National Insurance Number.
- **Composite Key**: A candidate key that consist of two or more attributes.

# Diagrammatic Representation of attributes



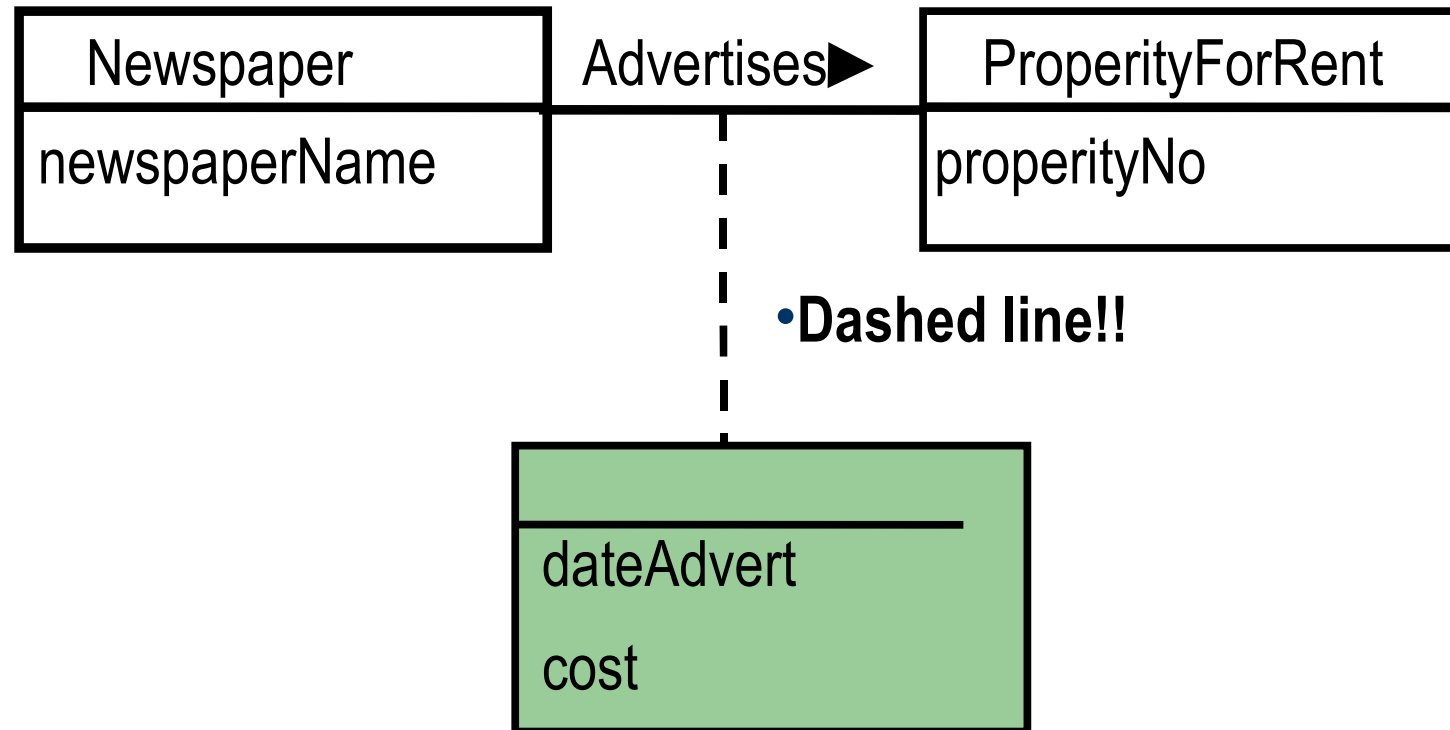
# ***Strong and Weak Entity Types***

- **Strong Entity Type**: An entity type that is not existence-dependent on some other entity type.
- **Weak Entity Type**: An entity type that is existence-dependent on some other type.



# ***Attributes on Relationships***

- Attributes can also be assigned to relationships.

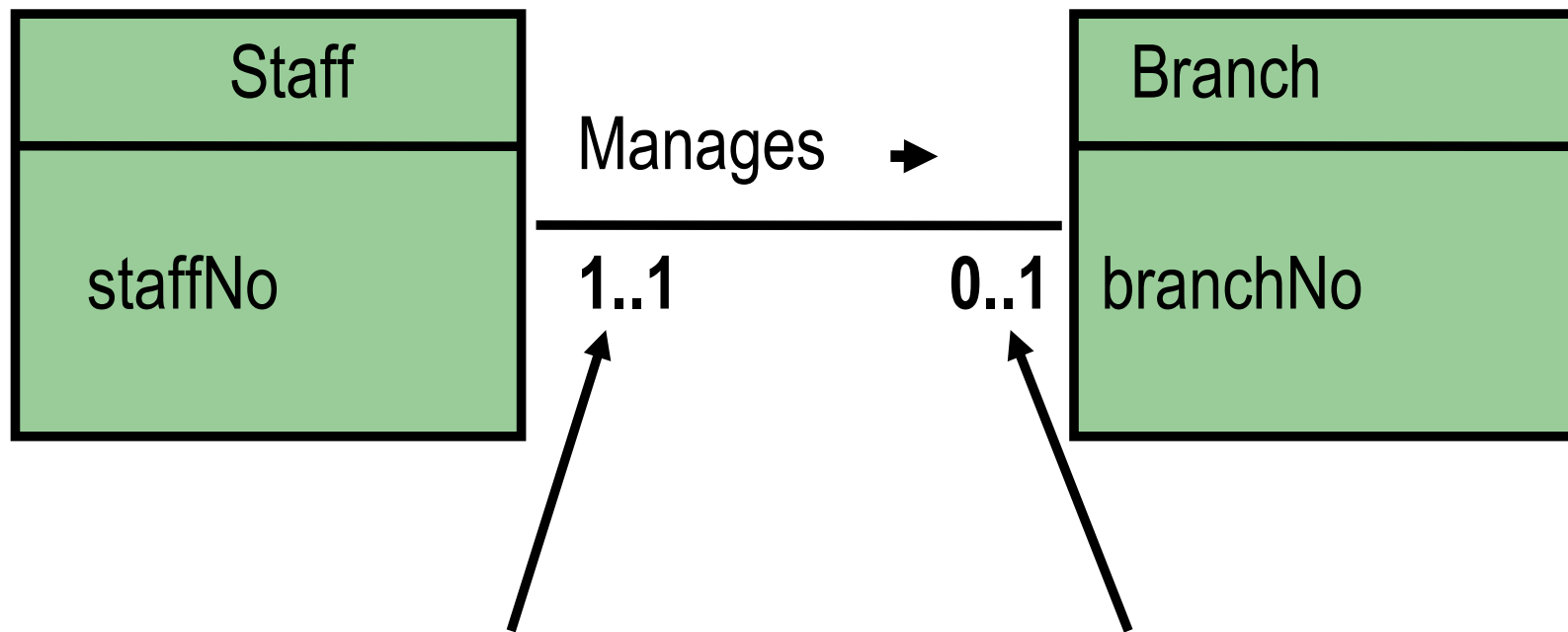




# ***Structural Constraints***

- **Multiplicity:** The number (or range) of possible occurrences of an entity type that may relate to a single occurrence of an associated entity type through a particular relationship.
  - **One-to-one (1:1)**
  - **One-to-many (1:\*)**
  - **Many-to-many (\*:\*)**

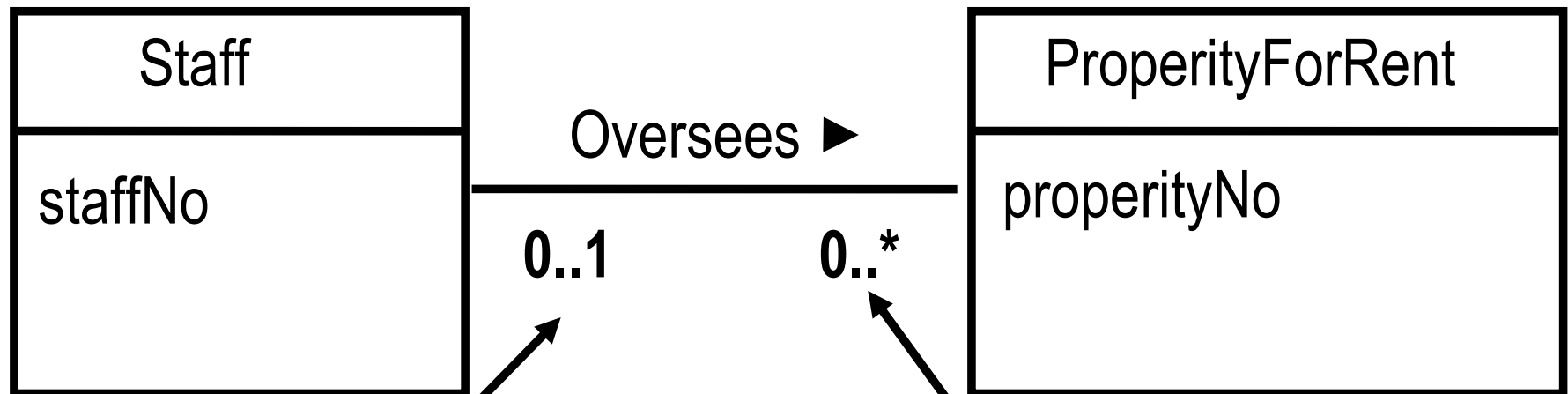
# One-to-One (1:1)



“Each branch **is managed** by  
One member of the staff”

“A member of staff **can manage**  
zero or one branch”

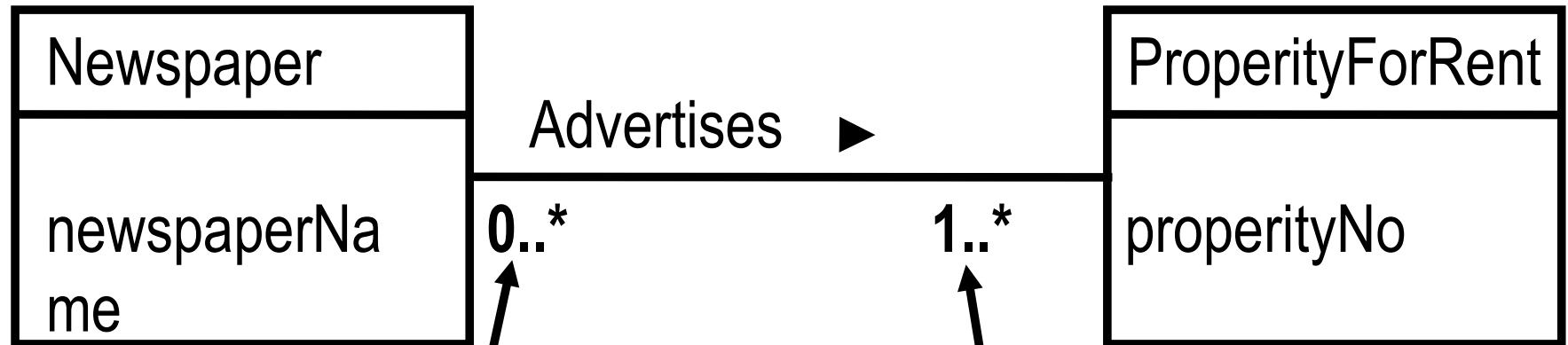
# One-to-Many (1:\*)



“Each property for rent **is overseen** by zero or one member of staff”

“Each member of staff **oversees** zero or more properitys for rent”

# Many-to-Many (\*:\*)

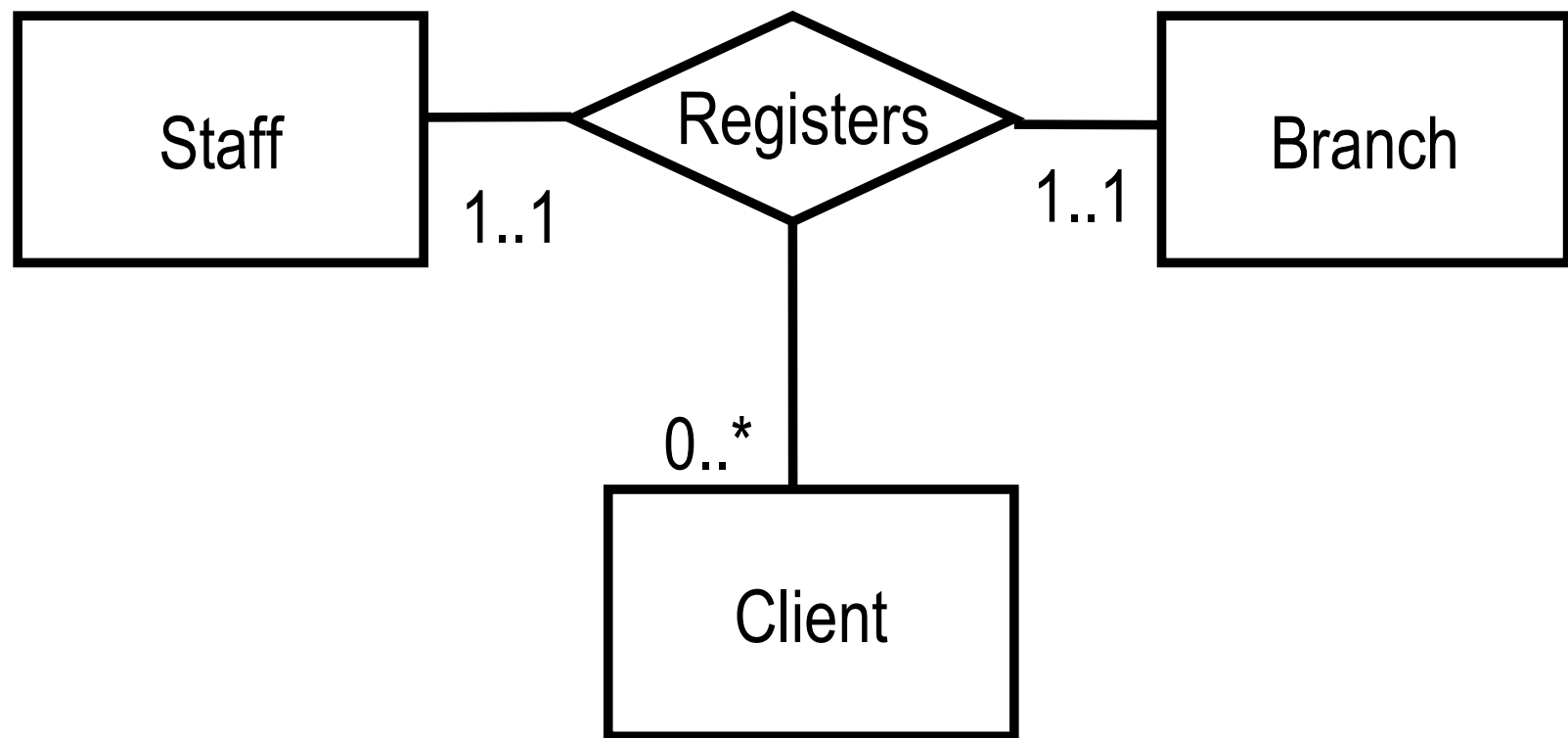


“Each property for rent **is advertised** in zero or more newspapers”

“Each newspaper **advertises** one or more properties for rent”

# ***Multiplicity for Complex Relationships***

- **Multiplicity** (complex relationships): The number (or range) of possible occurrences of any entity type in any n-ary relationship when the other (n-1) values are fixed.



# ***ERD: An Example***

- ✓ I need to maintain the information I use to contact people
- ✓ Maintain contact information
- ✓ Maintain some social information

## ***An Example: Address Book*** cont.

*“I use the address book to look up addresses and phone numbers of friends and businesses. I also contact some people by email and look at businesses’ Web sites. Some friends are on my holiday list, and I want to send them cards that are addressed with spouse and children’s names. I send some friends birthday cards.”*

## ***An Example : Address Book*** cont.

*“I use the address book to look up addresses and phone numbers of **friends** and **businesses**. I also contact some **people** by email and look at businesses’ Web sites. Some friends are on my **holiday list**, and I want to send them cards that are addressed with spouse and children’s names. I send some friends birthday cards.”*



# ***ERD: An Example*** cont.

---

***Noun = Entity***

***Verb = Relationship***

***Piece of Information = Attribute of an Entity***

# ***An Example: Address Book*** cont.

---

*Preliminary Entities:*

**Friends**

**Business**

**People**

**Holiday list**

# ***Address Book: Preliminary Attributes***

---

- Friends** • friendName
- Business** • friendAddress
- People** • friendPhone
- Holiday list** • friendFax
- friendEmail

# ***Address Book: Preliminary Attributes***

## ***Friends***

- friendName
- businessPhone

## ***Business***

- friendAddress
- businessFax

## ***People***

- friendPhone
- businessEmail

## ***Holiday list***

- friendFax
- businessURL
- friendEmail
- businessName
- businessContact
- businessContactPos

# ***Address Book: Preliminary Attributes***

## **Friends**

- friendName
- businessPhone

## **Business**

- friendAddress
- businessFax

## **People**

- friendPhone
- businessEmail

## **Holiday list**

- friendFax
- BusinessURL
- friendEmail
- Spouse
- businessName
- Kids Name
- businessContact
- birthday
- businessContactPos

# ***Address Book: Preliminary Attributes***

## ***Friends***

- friendName
- businessPhone

## ***Business***

- friendAddress
- businessFax

## ***People***

- friendPhone
- businessEmail

## ***Holiday* *list***

- friendFax
- BusinessURL
- friendEmail
- Spouse
- businessName
- Kids Name
- businessContact
- Birthday
- businessContactPos
- HolidayList

# Address Book: Entity & Attributes

## Friends

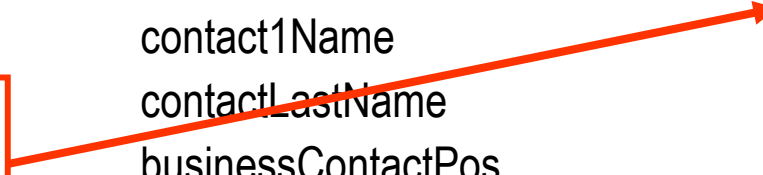
friend1stName  
friendLastName  
friendStreet  
friendCity  
friendState  
friendZIP  
friendAC  
friendPhoneNo  
friendFaxAC  
friendFaxNo  
friendEmail  
spouse  
kidnames  
birthMon  
birthDay  
holidayList

## Businesses

businessName  
contact1Name  
contactLastName  
businessContactPos  
businessStreet  
businessCity  
businessState  
businessZIP  
businessAC  
businessPhoneNo  
businessFaxAC  
businessFaxNo  
businessEmail  
businessURL

## Addresses

friendStreet  
friendCity  
friendState  
friendZIP



# Address Book: Entity & Attributes

## Friends

friend1stName  
friendLastName  
friendAC  
friendPhoneNo  
friendFaxAC  
friendFaxNo  
friendEmail  
spouse  
kidnames  
birthMon  
birthDay  
holidayList

## Businesses

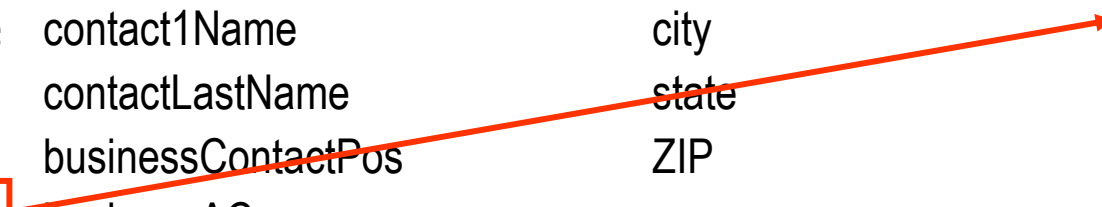
businessName  
contact1Name  
contactLastName  
businessContactPos  
businessAC  
businessPhoneNo  
businessFaxAC  
businessFaxNo  
businessEmail  
businessURL

## Addresses

street  
city  
state  
ZIP

## Telephones

friendAC  
friendPhoneNo





# ***Address Book: Entity & Attributes***

## **Friends**

friend1stName  
friendLastName  
friendAC  
friendPhoneNo  
friendFaxAC  
friendFaxNo  
friendEmail  
spouse  
kidnames  
birthMon  
birthDay  
holidayList

## **Businesses**

businessName  
contact1Name  
contactLastName  
businessContactPos  
businessAC  
businessPhoneNo  
businessFaxAC  
businessFaxNo  
businessEmail  
businessURL

## **Addresses**

street  
city  
state  
ZIP

## **Telephones**

friendAC  
friendPhoneNo  
phoneType



# Address Book: Entity & Attributes

## Friends

friend1stName  
friendLastName  
friendEmail  
spouse  
**kidnames**  
birthMon  
birthDay  
holidayList

## Businesses

businessName  
contact1Name  
contactLastName  
businessContactPos  
businessEmail  
businessURL

## Addresses

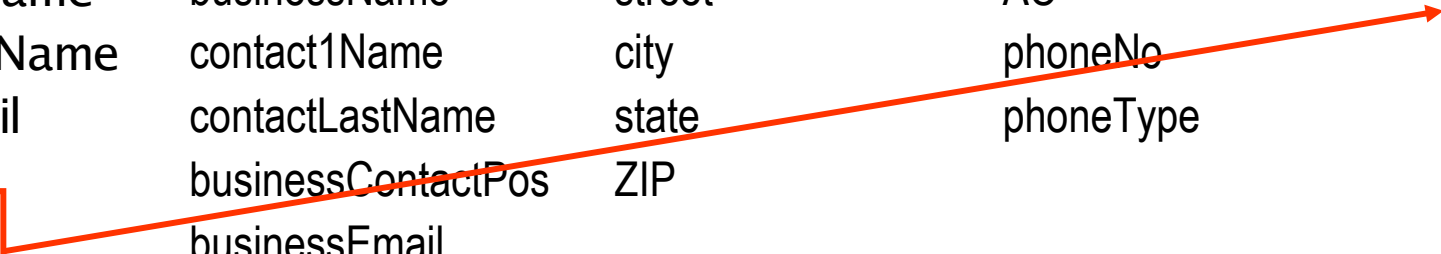
street  
city  
state  
ZIP

## Telephones

AC  
phoneNo  
phoneType

## Kids

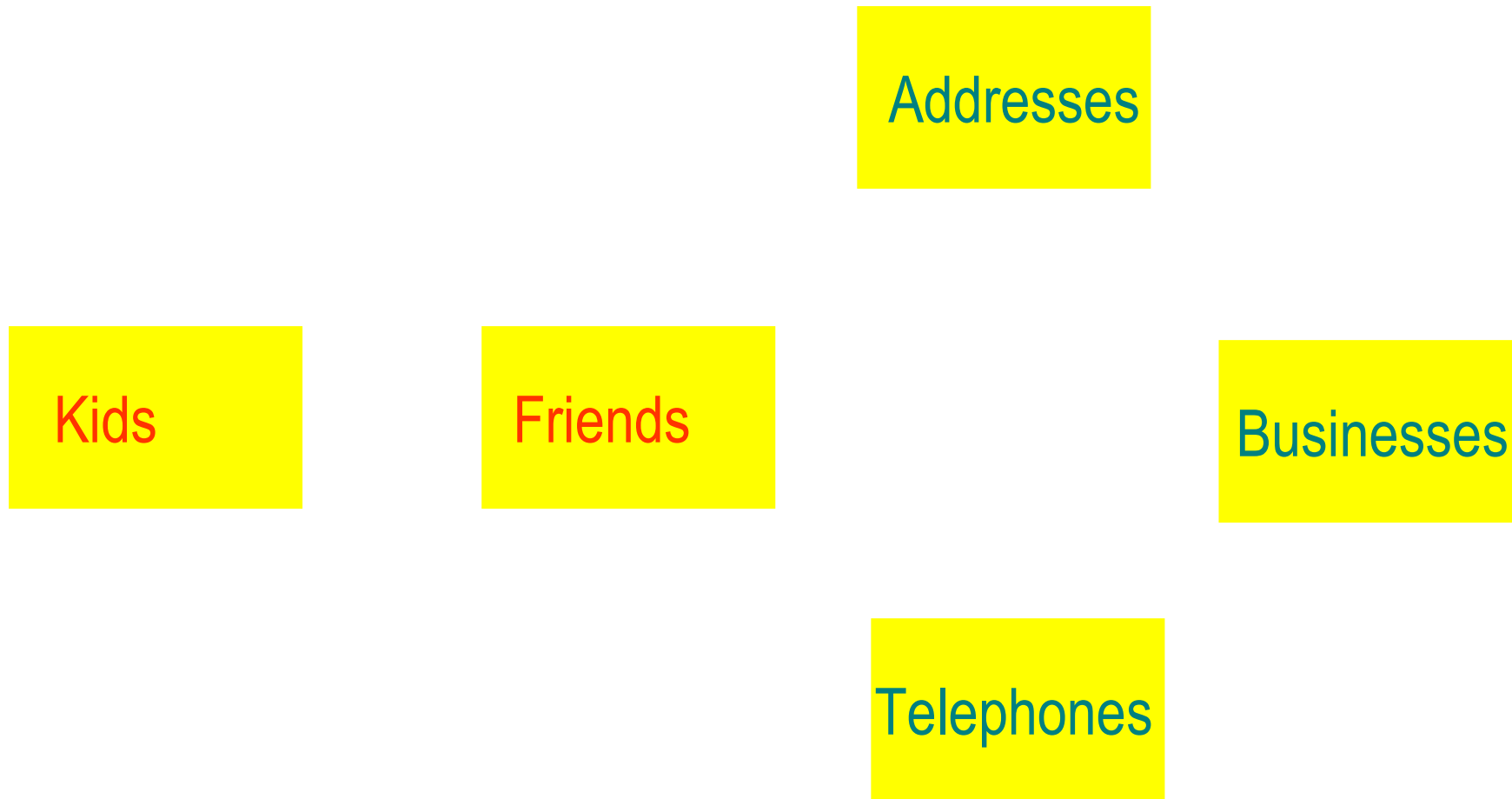
kidnames



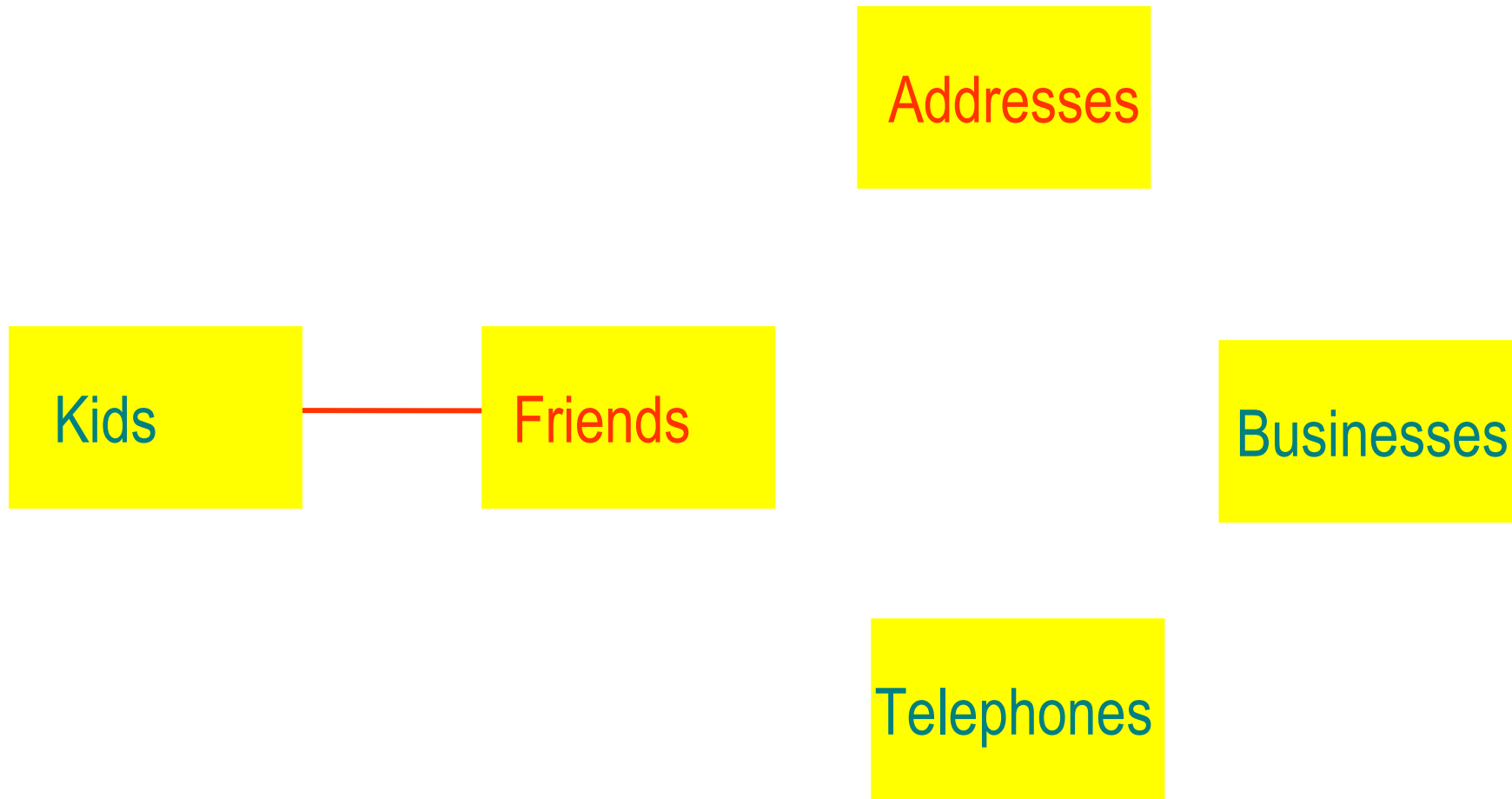
# Address Book: Entity & Attributes

Friends	Businesses	Addresses	Telephones	Kids
friend1stName	businessName	street	AC	kidnames
friendLastName	contact1Name	city	phoneNo	<b>*kidKey</b>
friendEmail	contactLastName	state	phoneType	
spouse	businessContactPosZIP		<b>*phonekey</b>	
birthMon	businessEmail	<b>*addkey</b>		
birthDay	businessURL			
holidayList	<b>*bkey</b>			
<b>*fkey</b>				

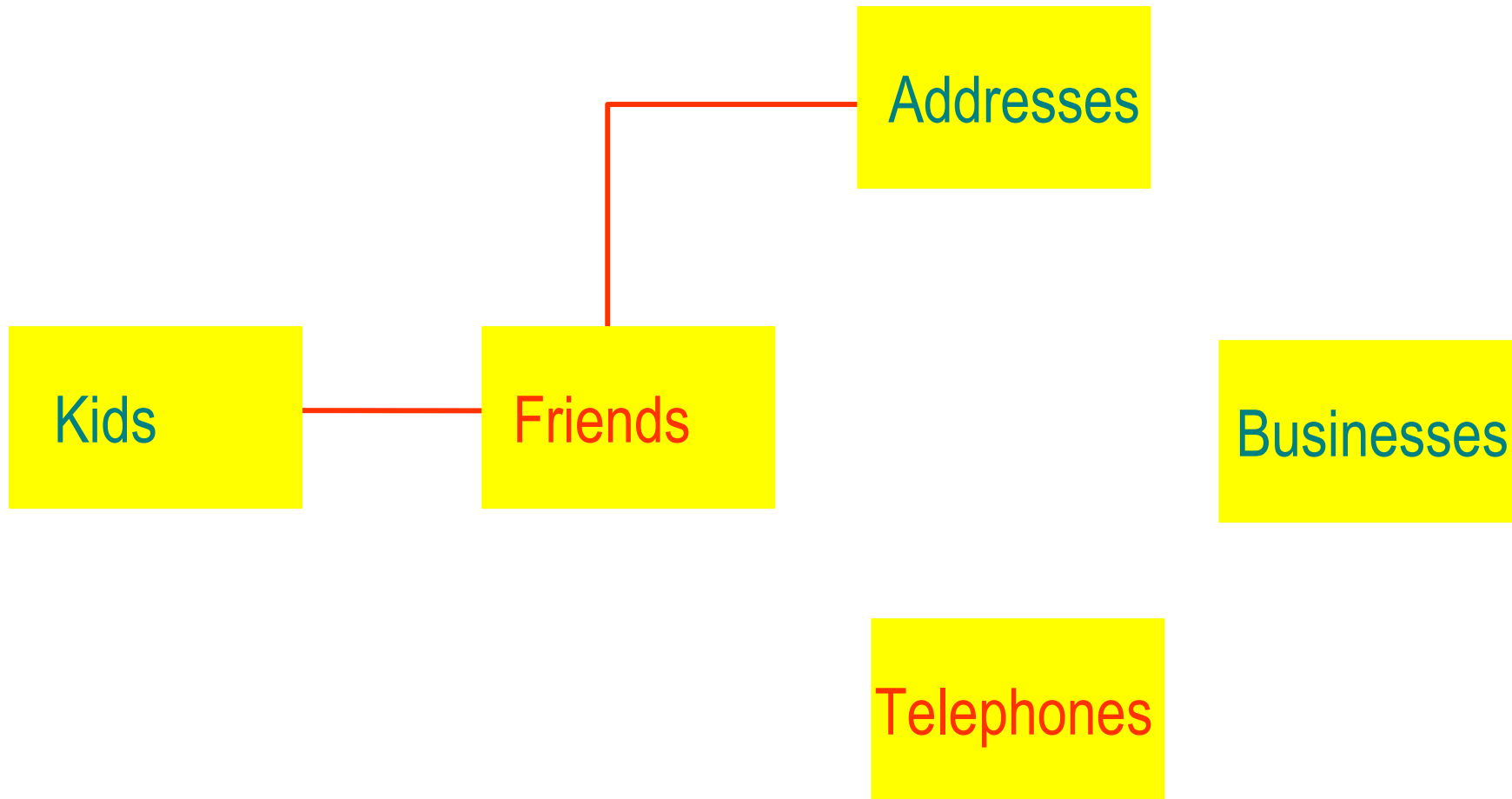
# ***Address Book: Entities***



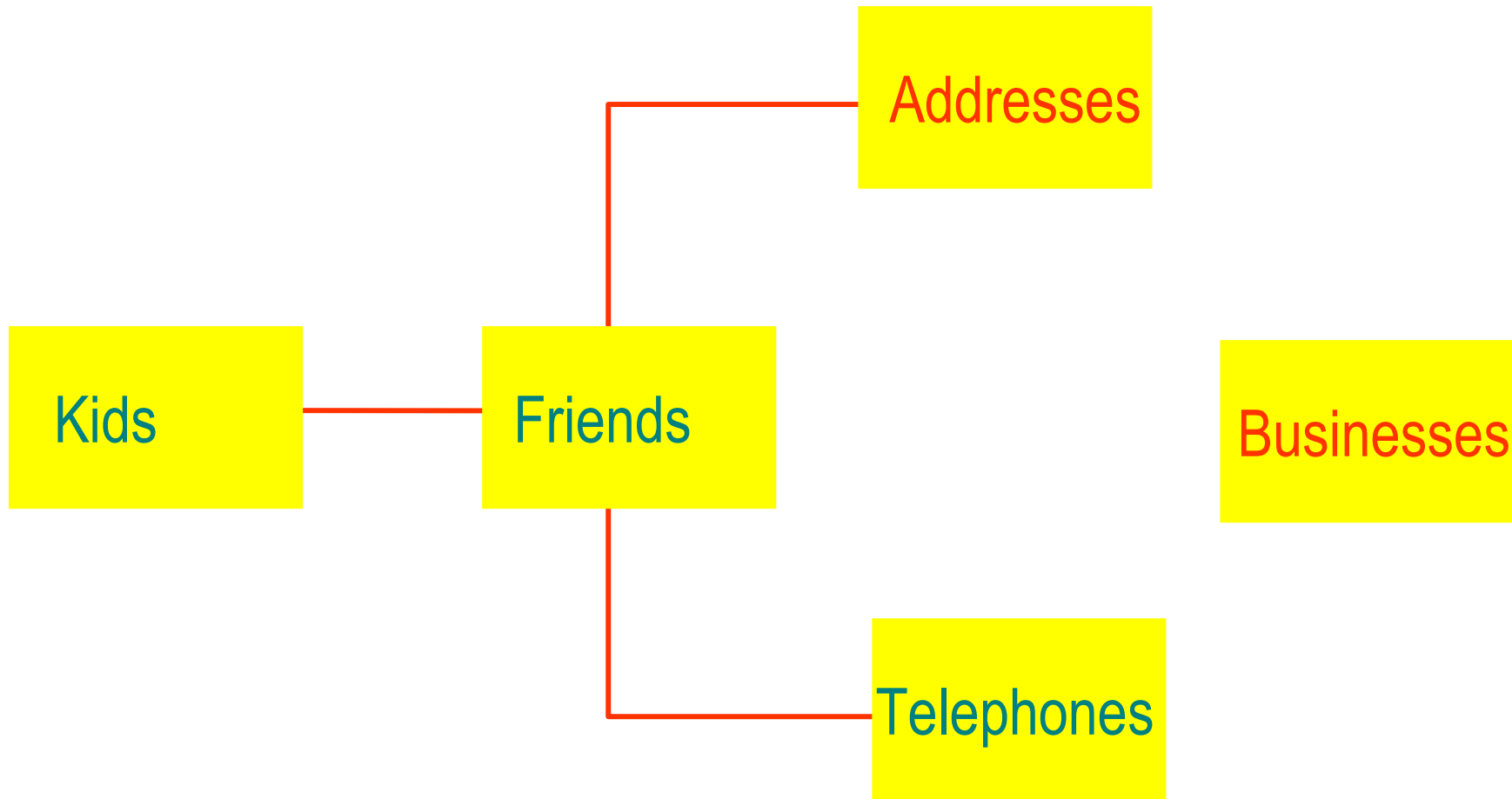
# ***Address Book: Entity Relationships***



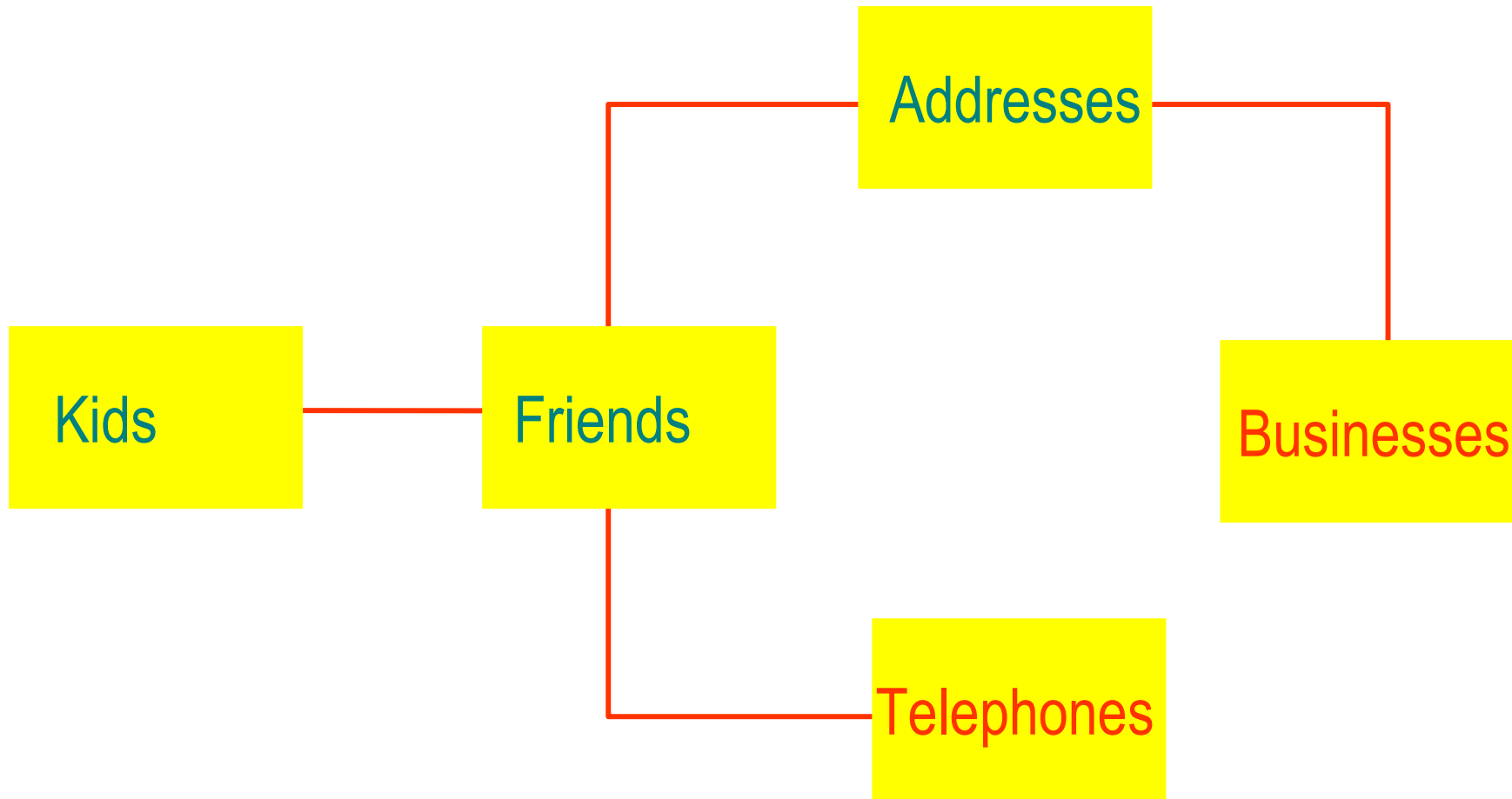
# ***Address Book: Entity Relationships***



# ***Address Book: Entity Relationships***

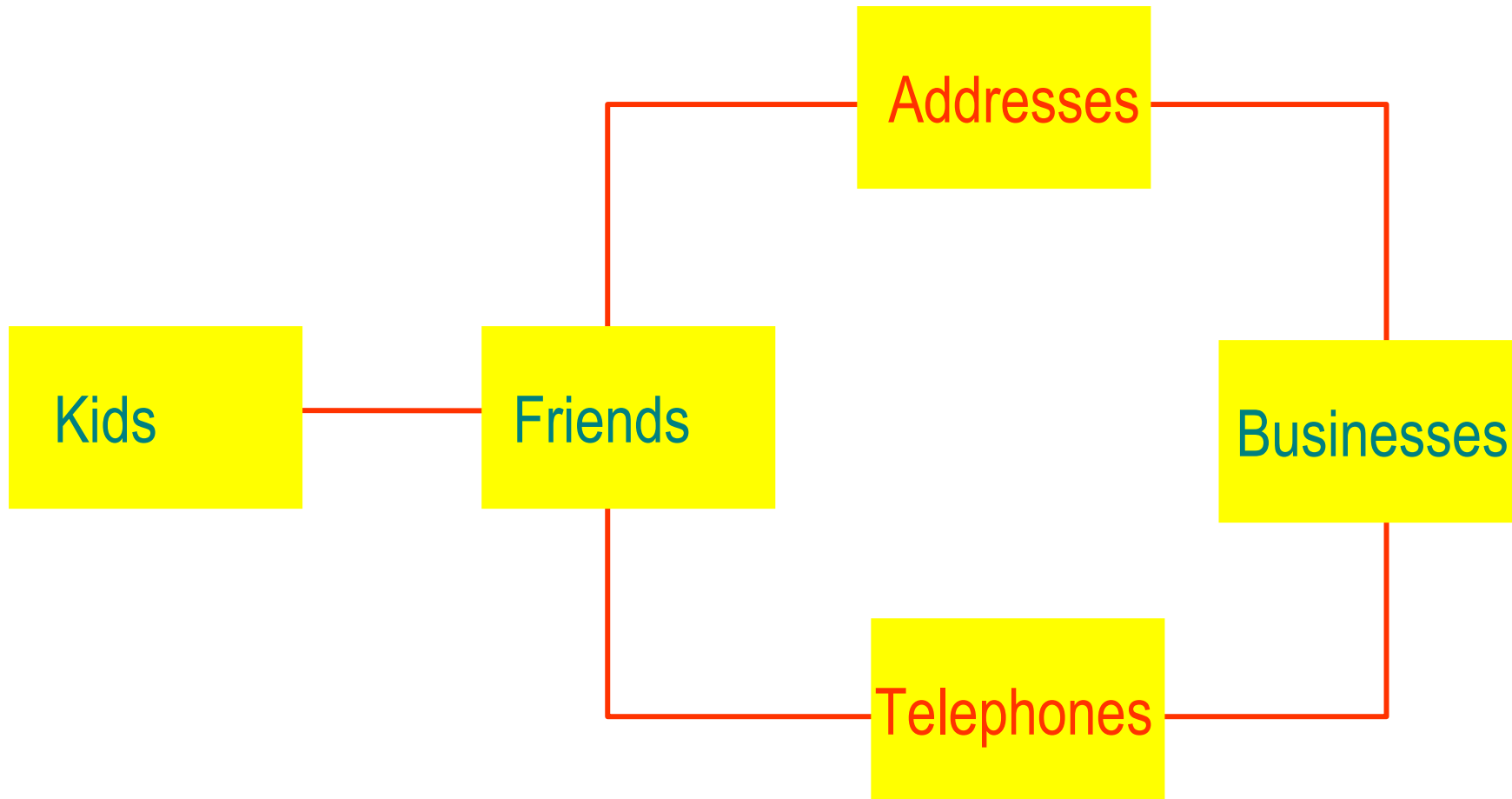


# ***Address Book: Entity Relationships***

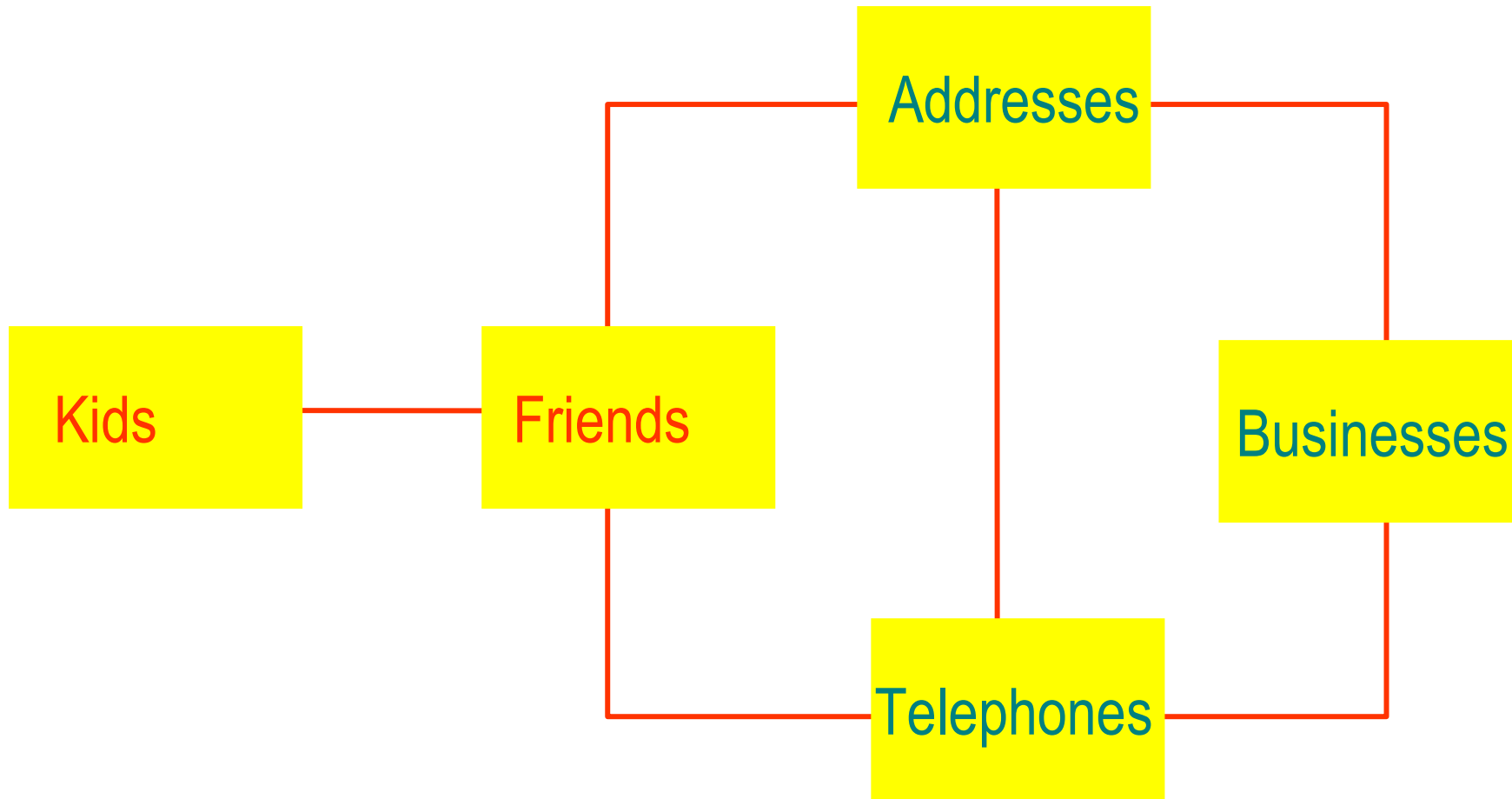




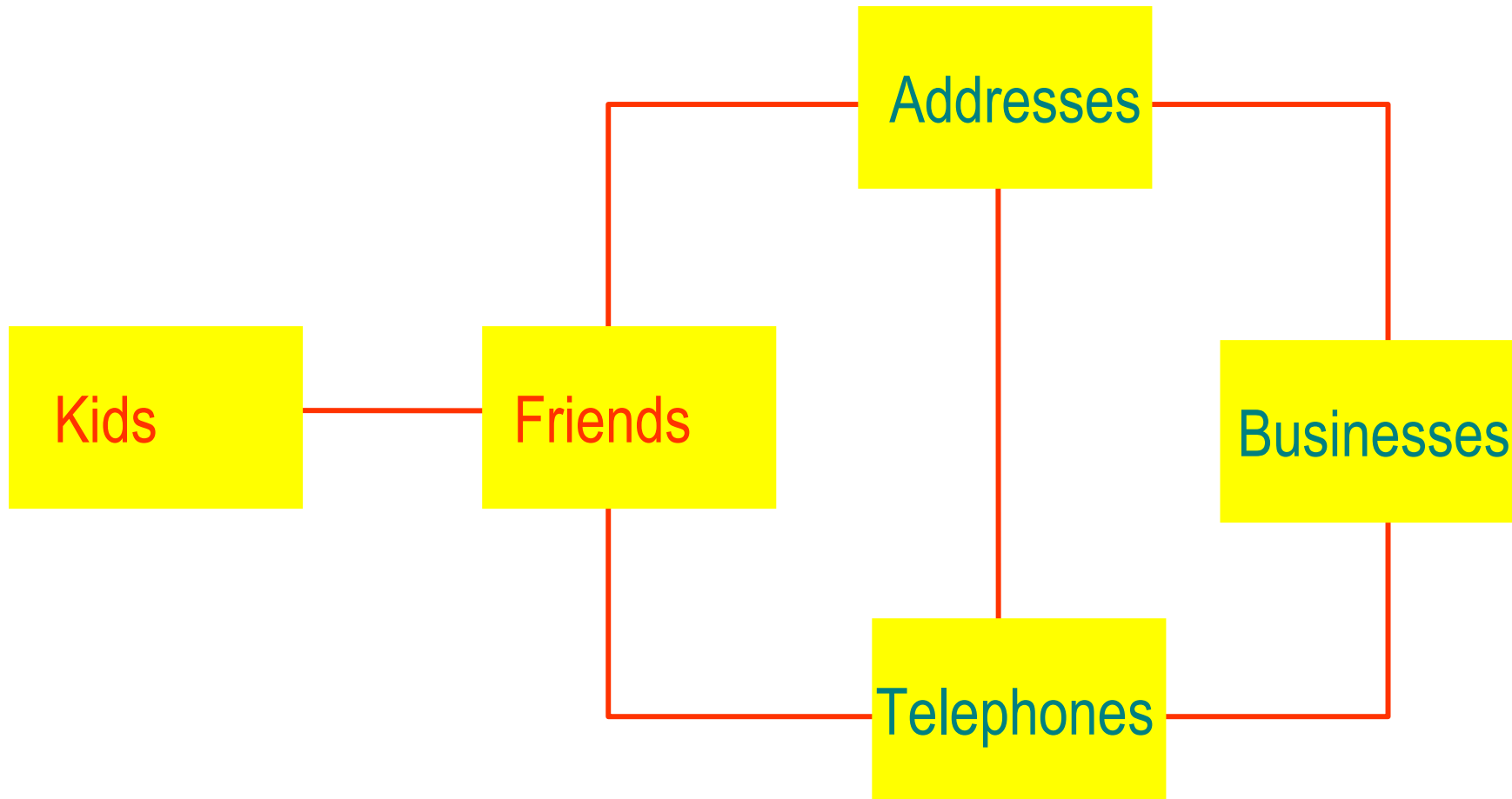
# ***Address Book: Entity Relationships***



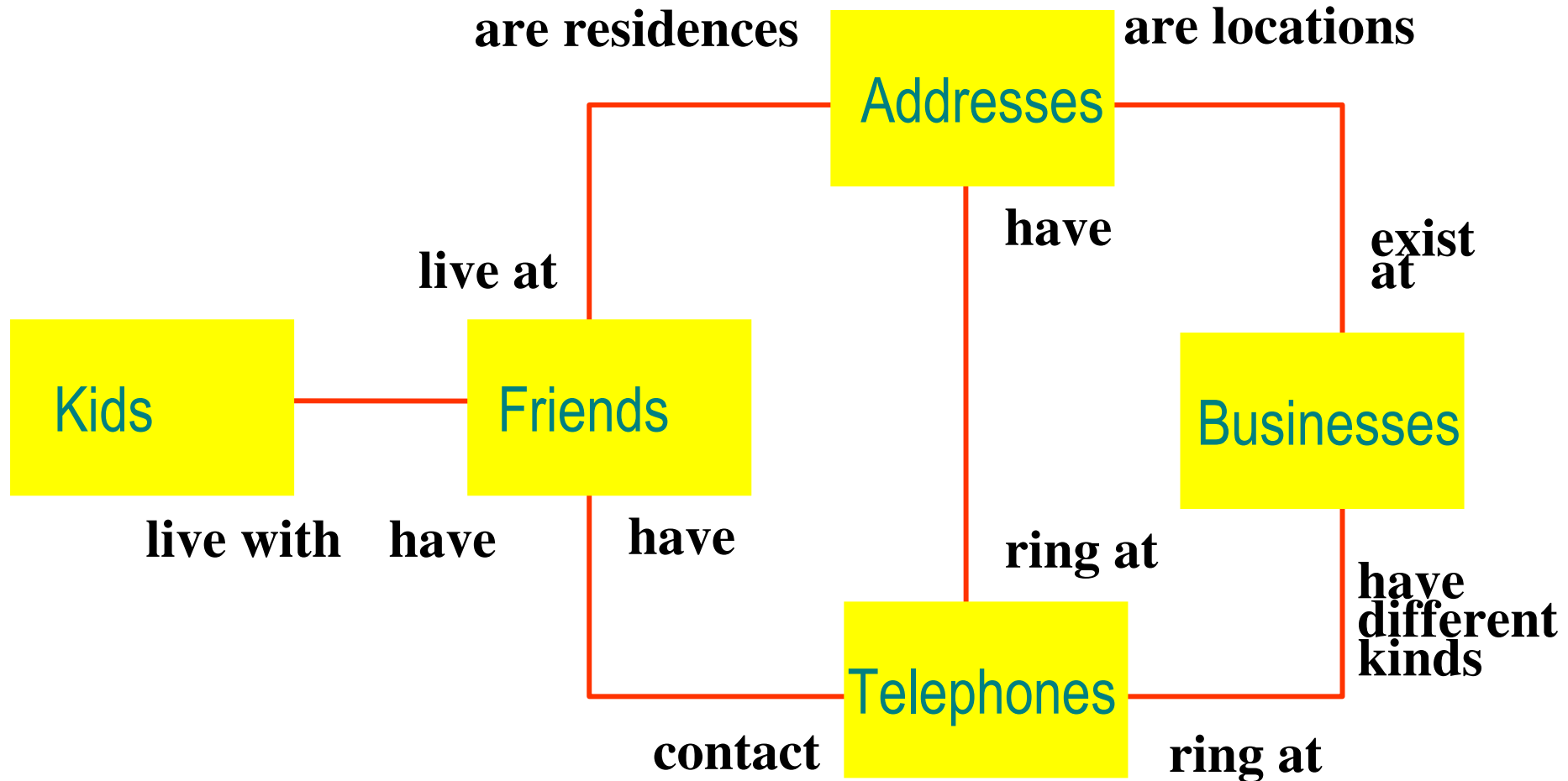
# ***Address Book: Entity Relationships***



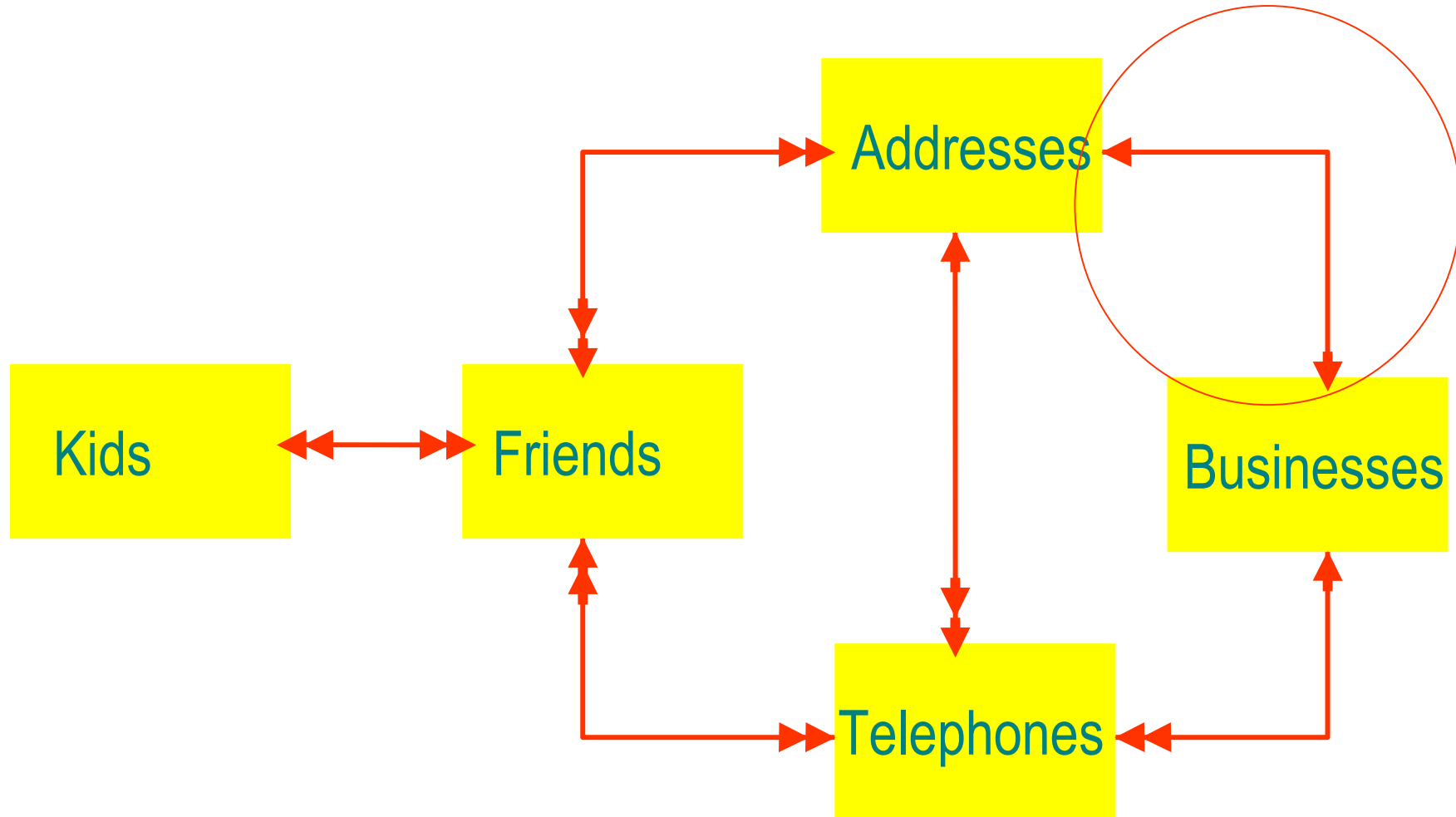
# ***Address Book: Entity Relationships***



# ***Address Book: Entity Relationships***



# ***Address Book: Entity Relationships***




# ***Rules for Foreign Key Assignment***

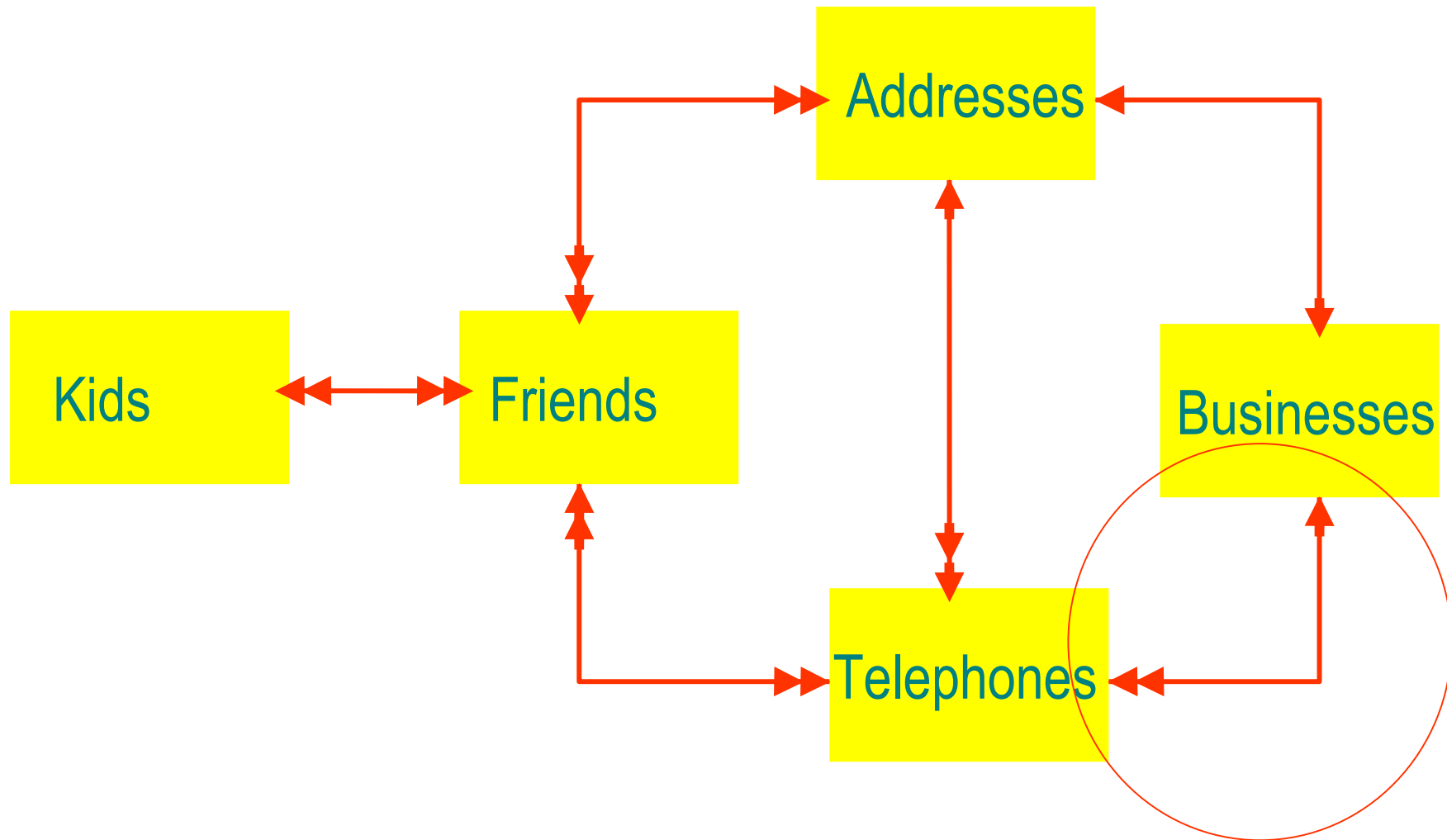
---

- **One-to-one:** Copy the primary key from the dominant table as a foreign key as subordinate table.
- **One-to-Many:** Copy the primary key from the “One” side as the foreign key into the “Many” side.
- **Many-to-Many:** Insert a third table between the two tables, and copy the primary key from each table into the new third table. The only field in this new table will be these foreign keys.

# Address Book: Entity Relationships

Friends	Businesses	Addresses	Telephones	Kids
friend1stName	businessName	street	AC	kidnames
friendLastName	contact1Name	city	phoneNo	<b>*kidKey</b>
friendEmail	contactLastName	state	phoneType	
spouse	businessContactPos	ZIP	<b>*phonekey</b>	
birthMon	businessEmail	<b>*akey</b>		
birthDay	businessURL	<b>+bkey</b>		
holidayList	<b>*bkey</b>			
<b>*fkey</b>				

# ***Address Book: Entity Relationships***

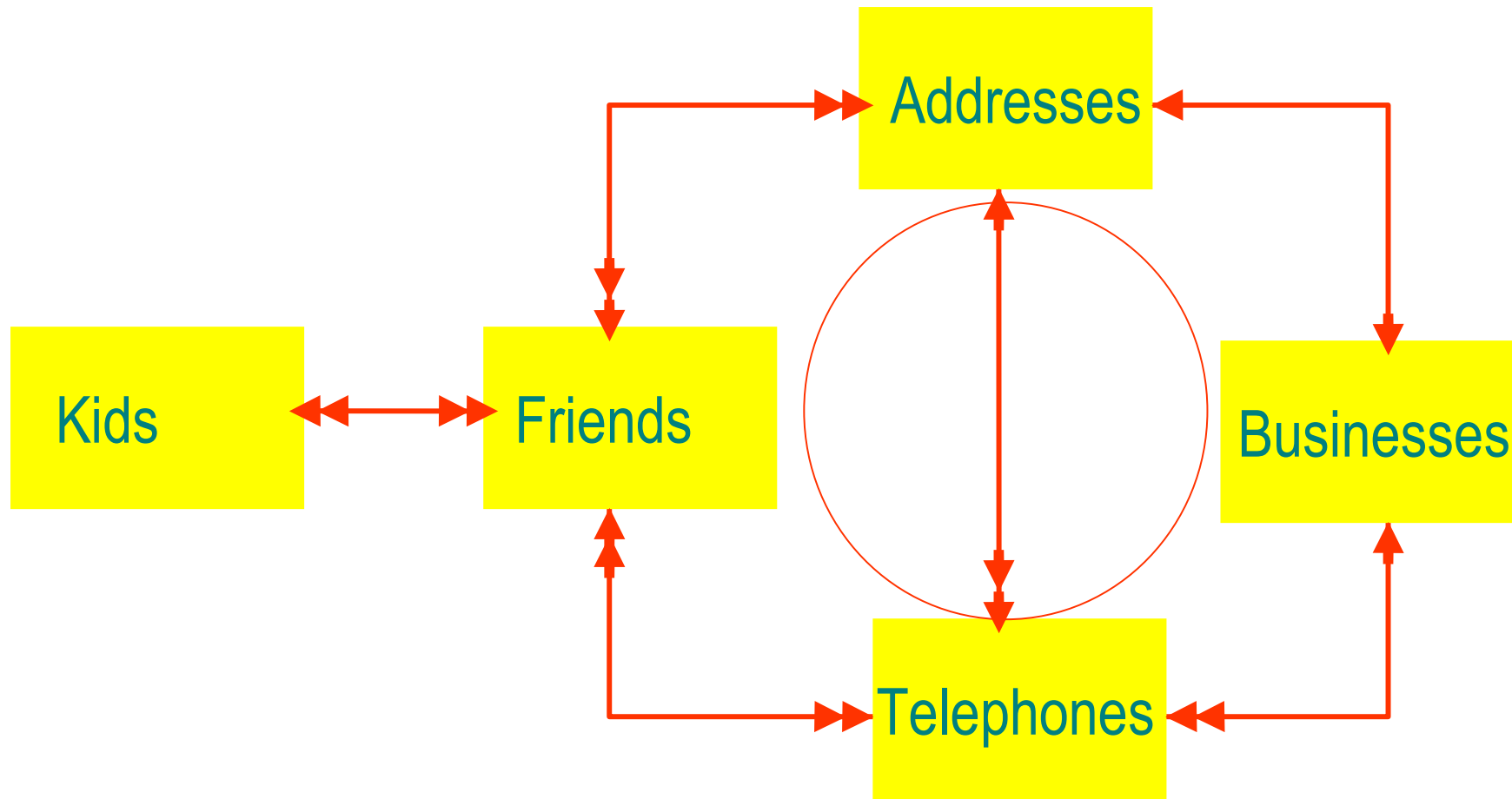




# Address Book: Entity Relationships

Friends	Businesses	Addresses	Telephones	Kids
friend1stName	businessName	street	AC	kidnames
friendLastName	contact1Name	city	phoneNo	<b>*kidKey</b>
friendEmail	contactLastName	state	phoneType	
spouse	businessContactPos	ZIP	<b>*phonekey</b>	
birthMonth	businessEmail	<b>*akey</b>	<b>+bkey</b>	
birthDay	businessURL	<b>+bkey</b>		
holidayList	<b>*bkey</b>			
<b>*fkey</b>				

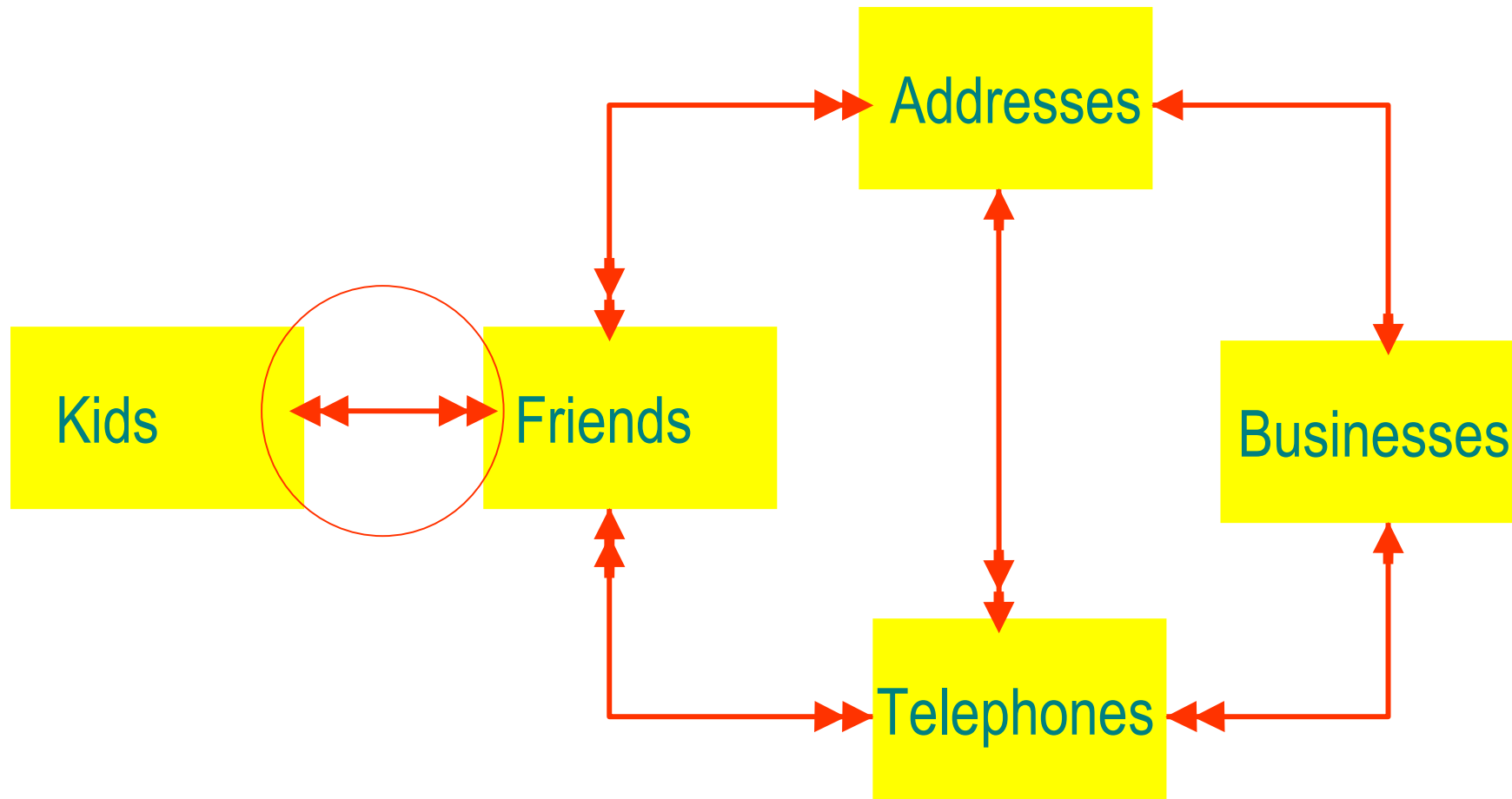
# ***Address Book: Entity Relationships***



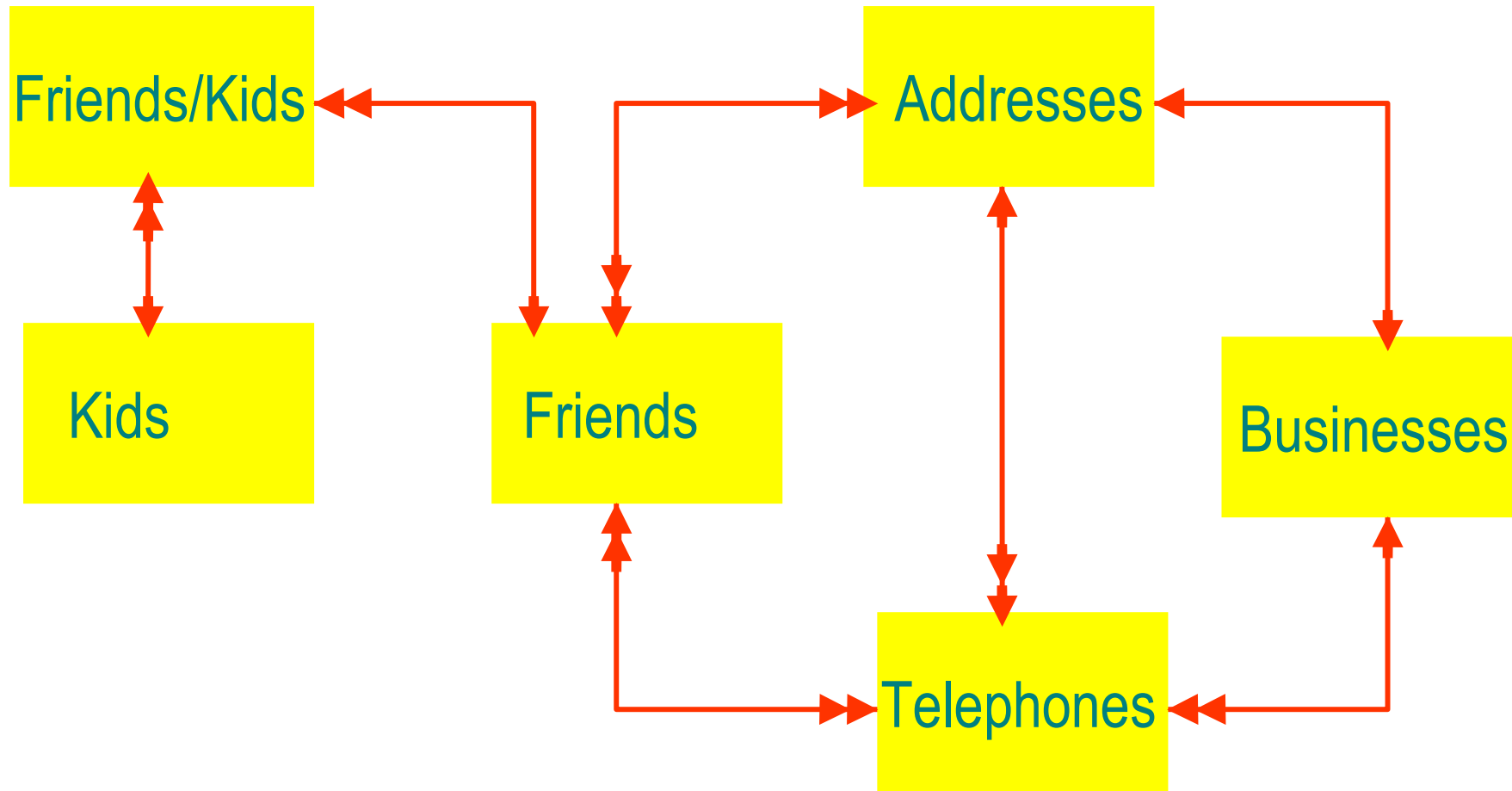
# Address Book: Entity Relationships

Friends	Businesses	Addresses	Telephones	Kids
friend1stName	businessName	street	AC	kidnames
friendLastName	contact1Name	city	phoneNo	<b>*kidKey</b>
e	contactLastName	state	phoneType	
friendEmail	businessContactPos	ZIP	<b>*phonekey</b>	
spouse	businessEmail	<b>*akey</b>	<b>+bkey</b>	
birthMon	businessURL	<b>+bkey</b>	<b>+akey</b>	
birthDay	<b>*bkey</b>			
holidayList				
<b>*fkey</b>				

# ***Address Book: Entity Relationships***



# ***Address Book: Entity Relationships***



# Address Book: Entity Relationships

## Friends

friend1stName  
friendLastNam  
e  
friendEmail  
spouse  
birthMon  
birthDay  
holidayList  
**\*fkey**

## Businesses

businessName  
contact1Name  
contactLastName  
businessContactPos  
businessEmail  
businessURL  
**\*bkey**

## Addresses

street  
city  
state  
ZIP  
**\*akey**  
**+bkey**

## Telephones

AC  
phoneNo  
phoneType  
**\*phonekey**  
**+bkey**  
**+akey**

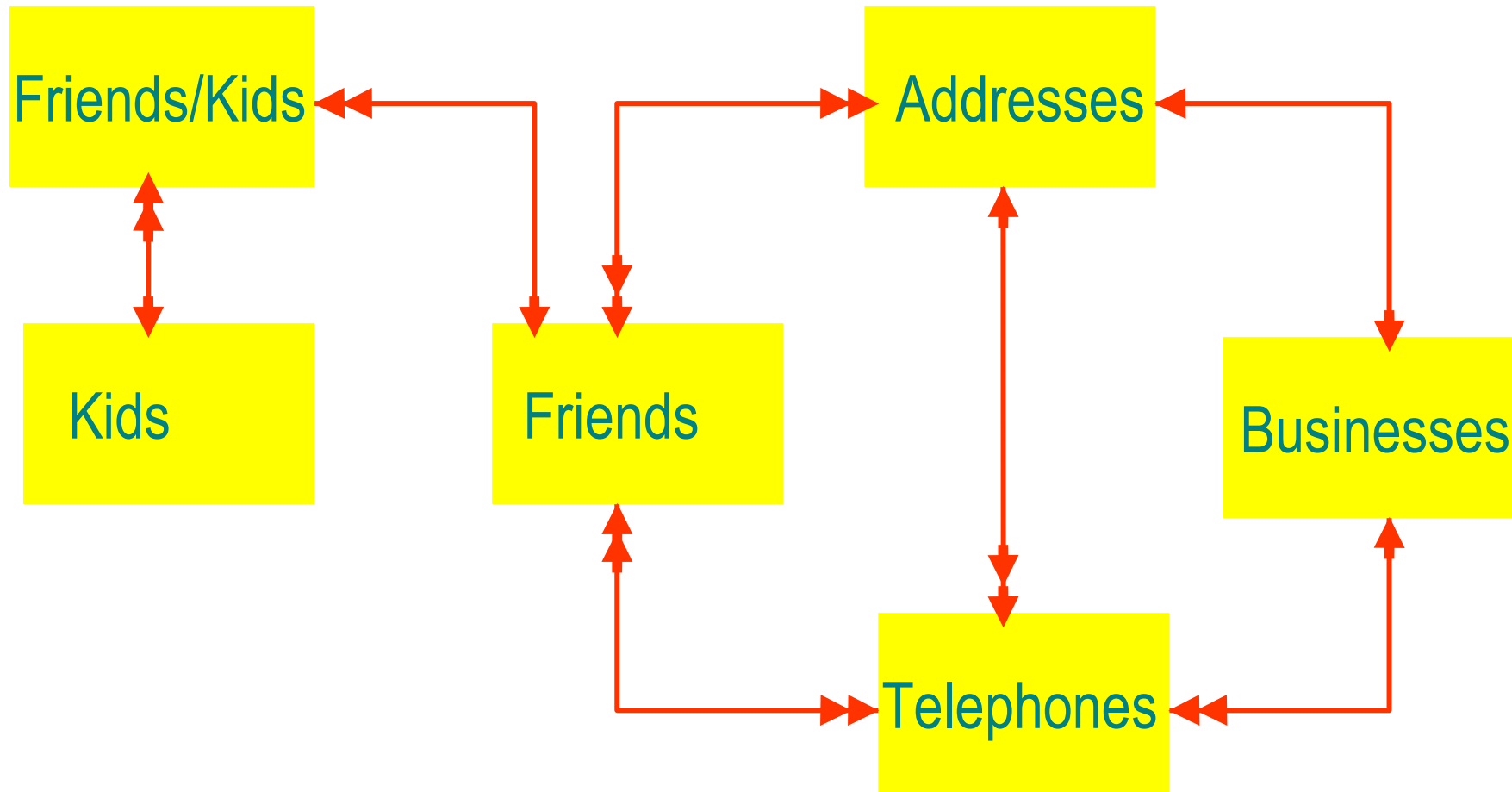
## Kids

kidnames  
**\*kidKey**

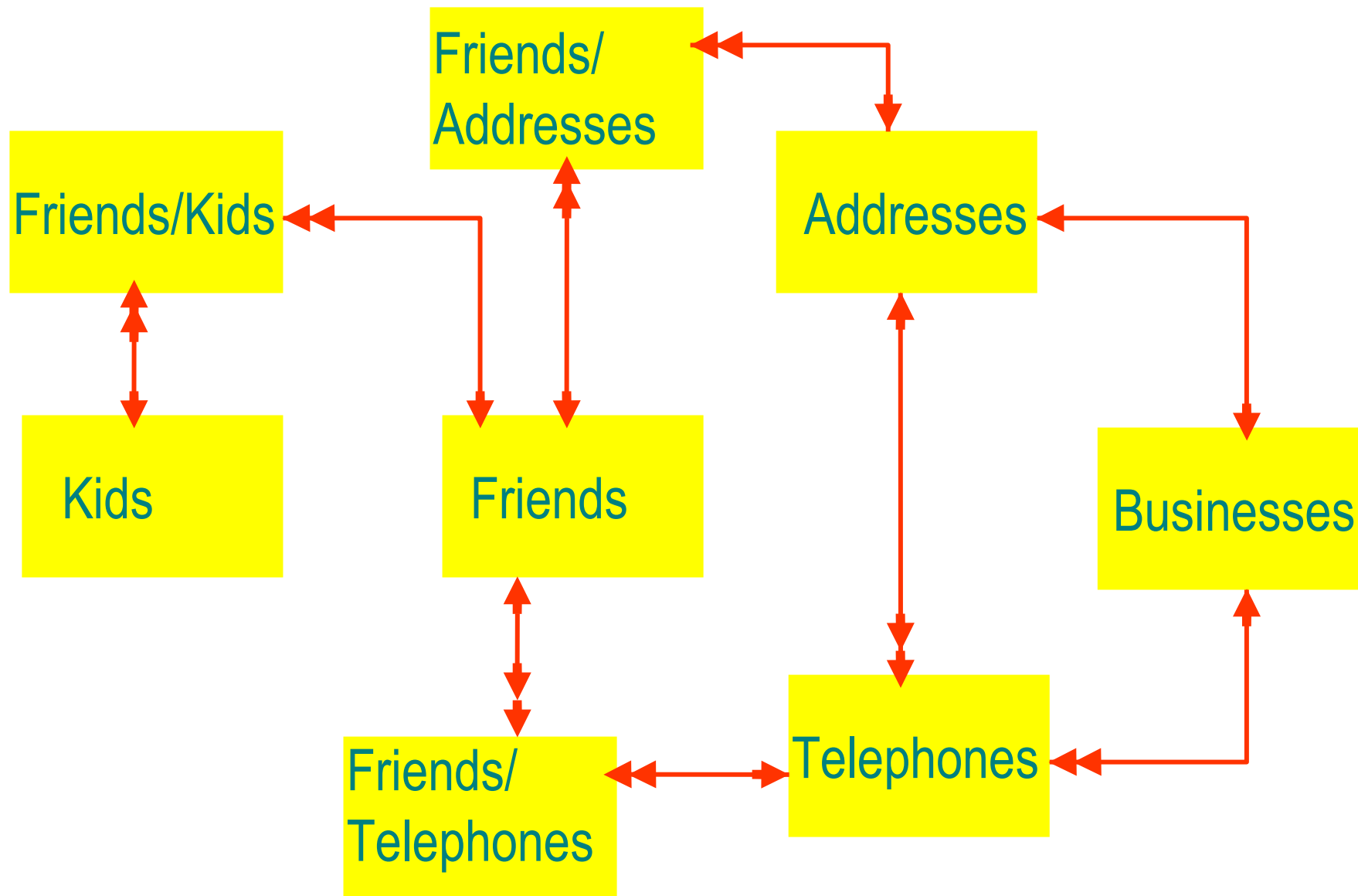
## Friends/Kids

+fkey  
+kidKey

# ***Address Book: Entity Relationships***



# ***Address Book: Entity Relationships***





# Address Book: Entity Relationships

Friends	Businesses	Addresses	Telephones	Kids
friend1stName	businessName	street	AC	kidnames
friendLastNam e	contact1Name	city	phoneNo	<b>*kidKey</b>
friendEmail	contactLastName	state	phoneType	
spouse	businessContactPos	ZIP	<b>*phonekey</b>	
birthMon	businessEmail	<b>*akey</b>	<b>+bkey</b>	
birthDay	businessURL	<b>+bkey</b>	<b>+akey</b>	
holidayList				

**\*fkey**

## Friends/Kids

+fkey  
+kidKey

## Friends/Telephones

+fkey  
+phonekey

## Friends/Addresses

+fkey  
+akey

# ***Summary***

---

From this lecture we have learned the details of

- ◆ Entities, attributes, relationships
- ◆ Different types of attributes
- ◆ Different Multiplicity constraints on entity relationships
- ◆ How to find entities, attributes, relationships
- ◆ How to deal with one-to-one, one-to-many, many-to-many relationships
- ◆ How to find primary and foreign keys