IACDM

Data Integration – Exercise Class 1

Data Integration

- *The aim of a **DIS** (Data Integration System) is to set up a system where it is possible to **query different data sources as** they were a **unique one**, through a **global schema**.*
- *To our aims, **two** types of **data integration**:*
  - ***Homogeneous data sources** → all the data sources use the same data model (e.g., all relational)*
  - ***Heterogeneous data sources** → different data models (e.g., one relational database to be integrated with an XML document)*

Phases of the integration of homogeneous data sources

1. **Source schema reverse engineering**
2. **Schema integration**
    a. Related concept identification + Conflict analysis and resolution
    b. Global conceptual schema
    c. Conceptual to logical translation of the global schema
3. **(GAV) Mapping** definition and **query answering**
    a. GAV Mapping definition
    b. Query formulation *[on the global schema]*
    c. Query rewriting

Source schema reverse engineering

- ***Given*** *the* ***logical schema*** *– that may also be not relational –,* ***find*** *the* ***ER schema***

Schema integration

- Related **concept identification** + **Conflict analysis** and **resolution** *(In this phase we draw a table putting in correspondence the different concepts in the different sources, and we decide how to solve the possible conflicts)*
- **Global conceptual schema** *(Here we produce the Entity-Relationship global schema, solving the conflicts as we have decided in the previous phase)*
- **Conceptual to logical translation** of the global schema *(In our exercises, we always use the relational model)*

(GAV) Mapping definition and query answering
GAV Mapping definition

- **GAV Mapping** definition
  (Define the relations of the global schema as views on the sources)
- **Query formulation** *[on the global schema]*
- **Query rewriting**
  (In this phase the queries are rewritten in terms of the views using the mappings)
  [Note that usually in the exercises we will start with the query on the global schema, because the exercises usually ask to define the GAV mappings just for the tables involved in the query answer.]

Conflicts

Types of conflicts:

- **Name conflicts** *(i.e., synonimies and homonimies: e.g., client vs customer)*
- **Type conflicts** *(In parentheses because usually we do not consider them in our exercises. For example, if an identifier is a string in one case and an integer in another case)*
- **Data semantics** *(E.g., different currencies or unit of measure)*
- **Structure conflicts** *(E.g., a concept is an attribute in a source and an entity in another one)*
- **Cardinality conflicts** *(E.g., a movie may have just one director in a source, while another source allows more directors)*
- **Key conflicts** (*E.g., in a source the person is identified by the SSN and in another source is identified through the email).*

Exercise

LALuxuryHouses is a real estate agency located in **Los Angeles** and its business is **exclusively focused** on **luxury villas** located in the Los Angeles area (State of California).

Differently, **USAHouses** is an important real estate agency that **rents** and **sells houses** in all the main states of the USA.

USAHouses wants to **increase** its **business** in **Los Angeles**. Since the Los Angeles area is currently only partially covered by the agencies of USAHouses, its management decided to **buy LALuxuryHouses** and founded a **new company** called **USARealEstateCompany**. The management of USARealEstateCompany (the new company) wants to **integrate** the **information** available in the **two sources** (LALuxuryHouses and USAHouses) in order to be able to query all the available data.

LALuxuryHouses

CLIENTS (SSN, Lastname, Firstname, Address, City, State, Age, PhoneNumber)

EMPLOYEE (IDEmployee, Lastname, Firstname, PhoneNumber)

HOUSES (HouseAddress, HouseCity, SizeSquareMeters, Rooms) // The size of each home is measured in square meters.

HOUSE-OWNEDBY (HouseAddress, HouseCity, ClientSSN) // Table House-OwnedBy is used to store the information about the owners of each house.

RENTAL-CONTRACT (IDRentContract, HouseAddress, HouseCity, StartDate, EndDate, AnnualCost, IDEmployee) // Each tuple in Table Rental-Contract represents the rental of a house (identified by the pair HouseAddress, HouseCity) for the period from StartDate to EndDate

RENTEDBY (IDRentContract, ClientSSN) // Table RentedBy is used to store who are the clients associated to each rental contract (i.e., who rented the house associated to the contract).

SALE (IDSaleContract, HouseAddress, HouseCity, Date, Cost, IDEmployee) // Each tuple in Sale corresponds to one sale.

SOLDTO (IDSaleContract, ClientSSN) // Table SoldTo is used to store who are the buyers associated to each sale.

USAHouses

BUYERS (BuyerID, Name, Surname, Address, City, State, YearOfBirth, SSN, PhoneNumber) // Each tuple in Table Buyers represents someone who bought or rented a real estate

OWNERS (OwnerID, Name, Surname, Address, City, State, YearOfBirth, SSN, PhoneNumber) // Each tuple in Table Owners represents someone who owns a real estate

AGENTS (AgentID, Name, Surname, MobilePhoneNumber, OfficePhoneNumber)

REALESTATES (IDRE, Address, City, State, NumOfRooms, Size_SquareFeet, NumberOfFloors, OwnerID) // The size of each real estate is measured in square feet.

REALESTATE-RENTAL (IDRE, StartDate, EndDate, BuyerID, AgentID, MonthlyCost)

REALESTATE-SALE (IDRE, Date, BuyerID, AgentID, Price)

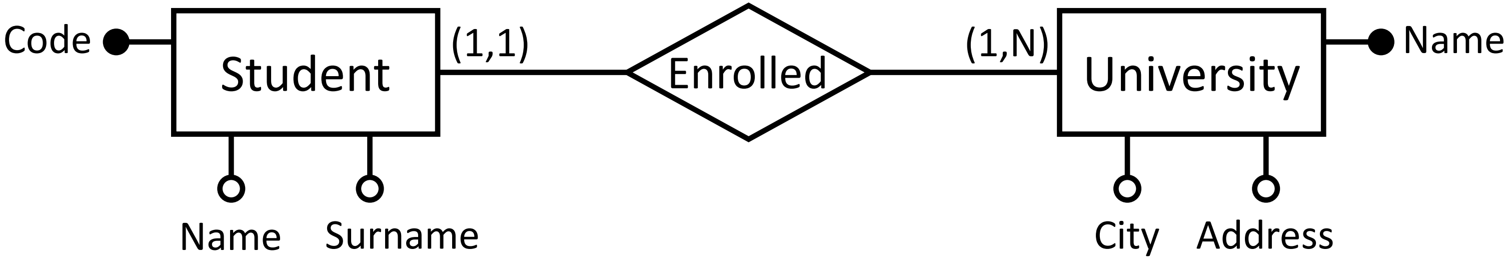
Source schema reverse engineering

Provide, for each input data source, the reverse engineering from the logical to the conceptual schema (ER graph)

- Tables in the logical schema normally represent either:
    - Entities
    - N:N relationships
    - Weak entities
- In the logical schema, foreign keys are always on the side of the "1" in a 1:N relationship.
- Cardinalities for the attributes must be specified when different form (1,1).

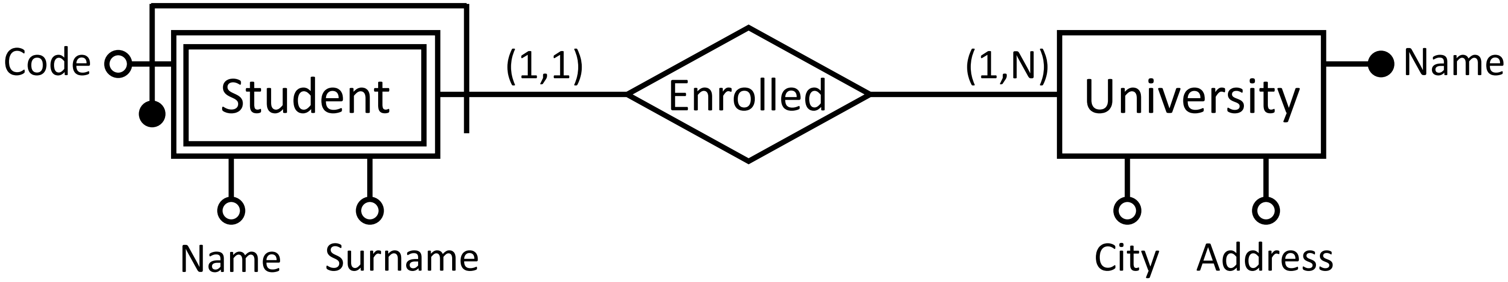
Weak Entities

Sometimes the attributes of an entity are not enough to identify uniquely its records:



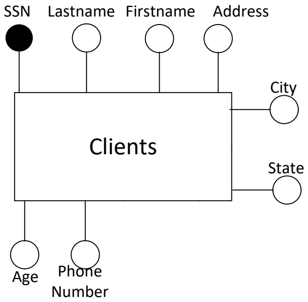
Weak Entities

Sometimes the attributes of an entity are not enough to identify uniquely its records:



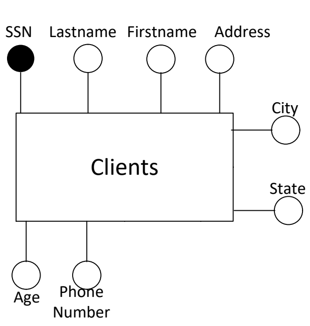
LALuxuryHouses

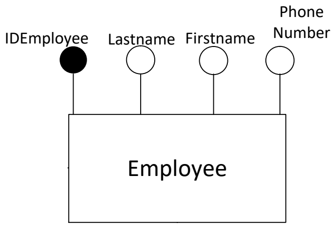
CLIENTS (<u>SSN</u>, Lastname, Firstname, Address, City, State, Age, PhoneNumber)



LALuxuryHouses

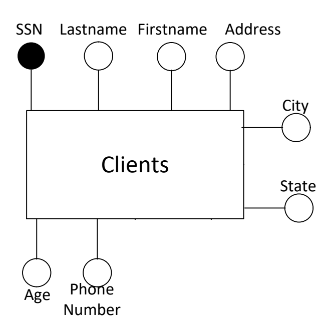
EMPLOYEE (IDEmployee, Lastname, Firstname, PhoneNumber)

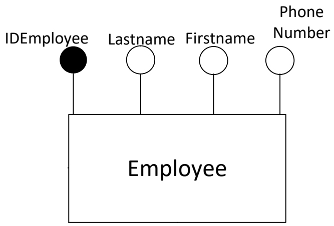


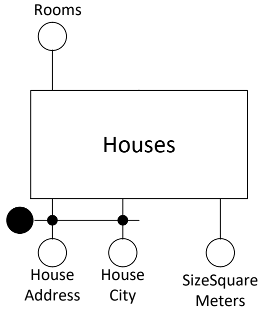


LALuxuryHouses

HOUSES (HouseAddress, HouseCity, SizeSquareMeters, Rooms)

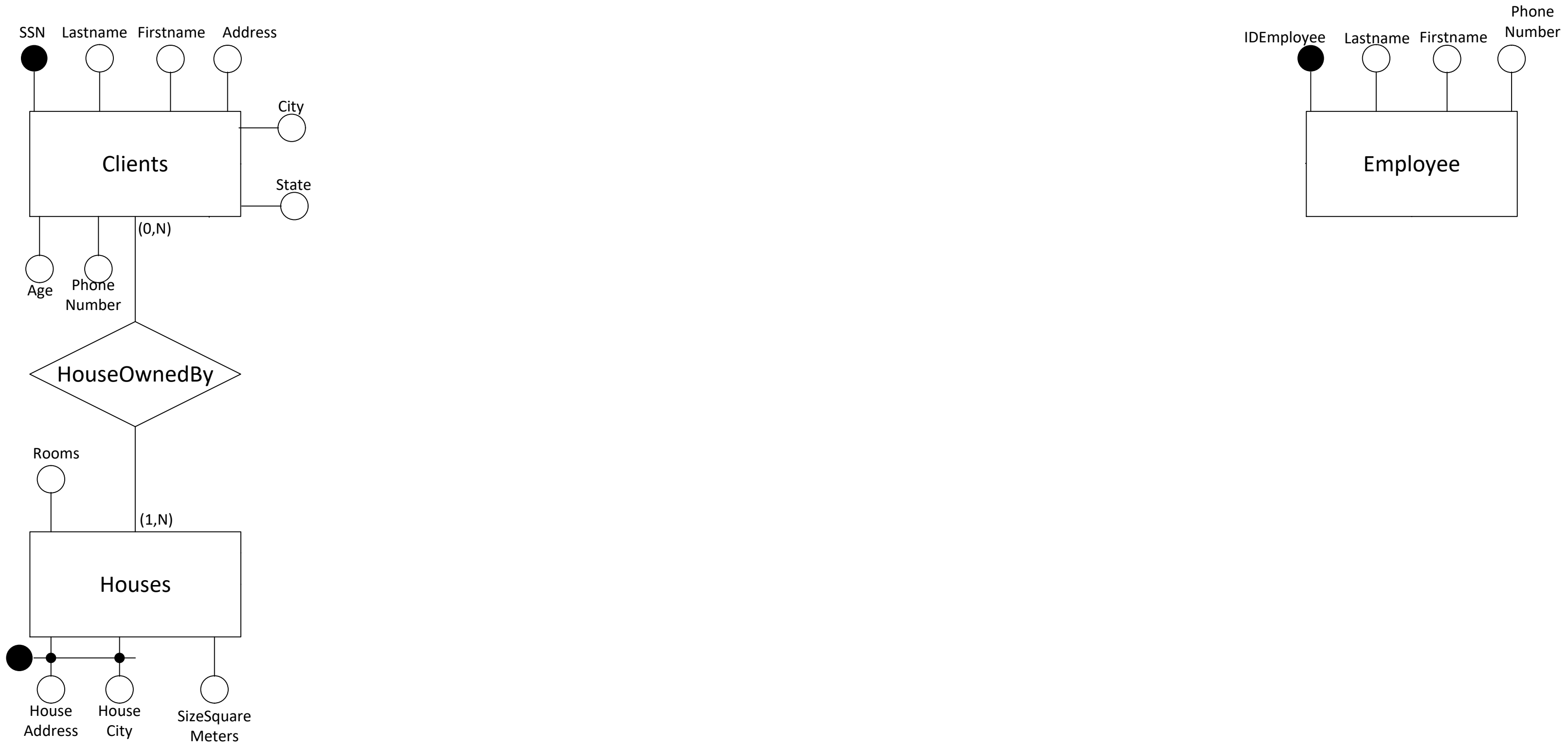






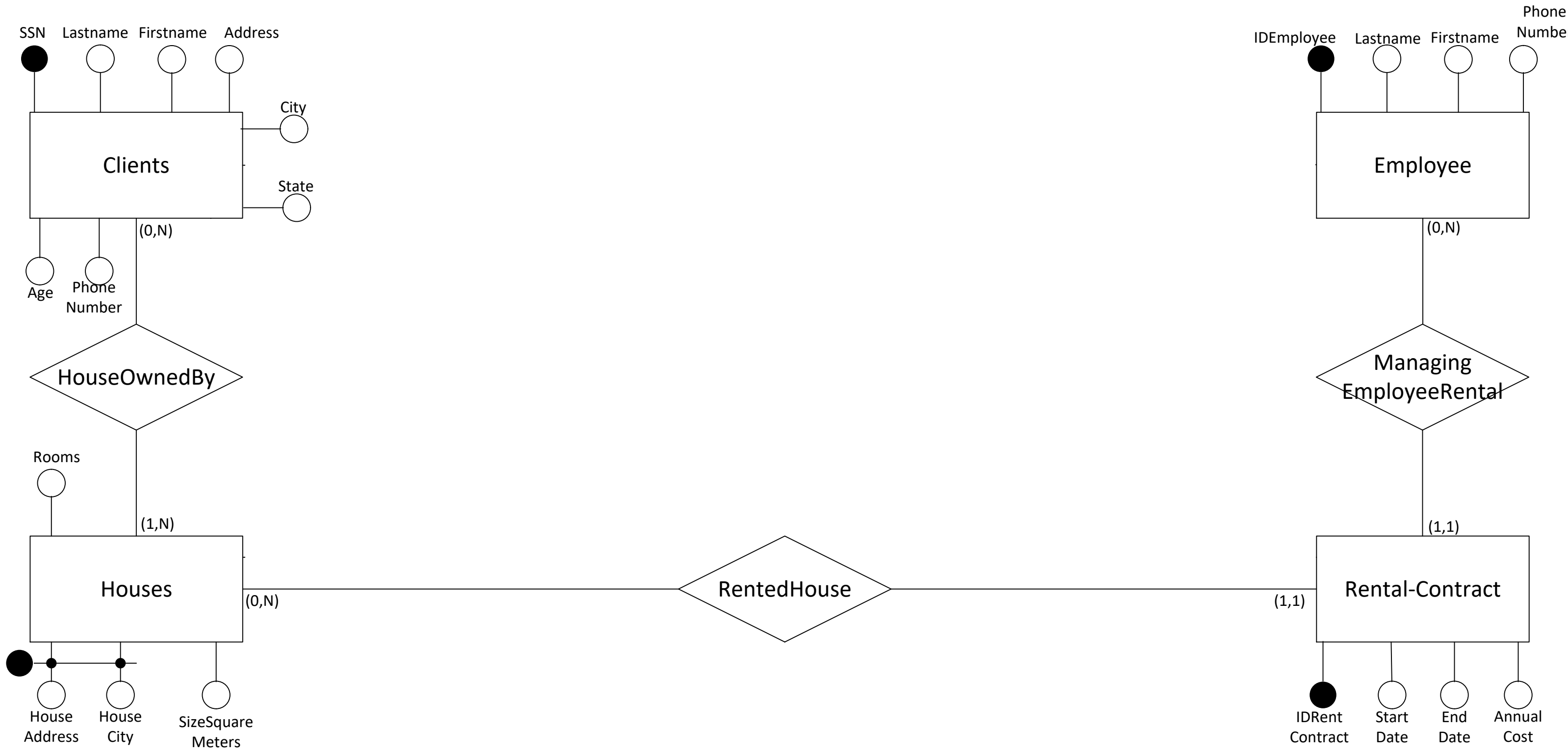
LALuxuryHouses

HOUSE-OWNEDBY (HouseAddress, HouseCity, ClientSSN)



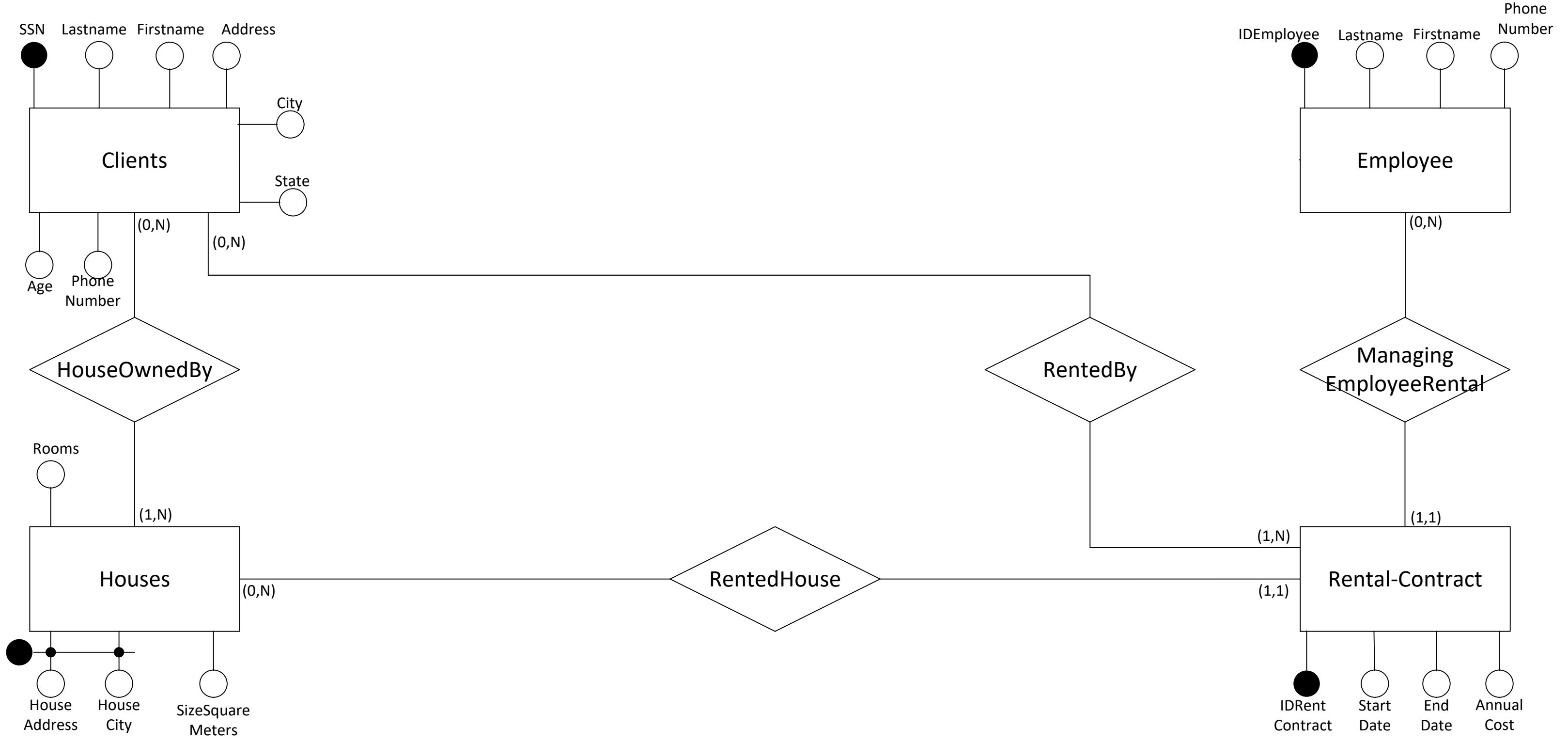
LALuxuryHouses

RENTAL-CONTRACT (IDRentContract, HouseAddress, HouseCity, StartDate, EndDate, AnnualCost, IDEmployee)



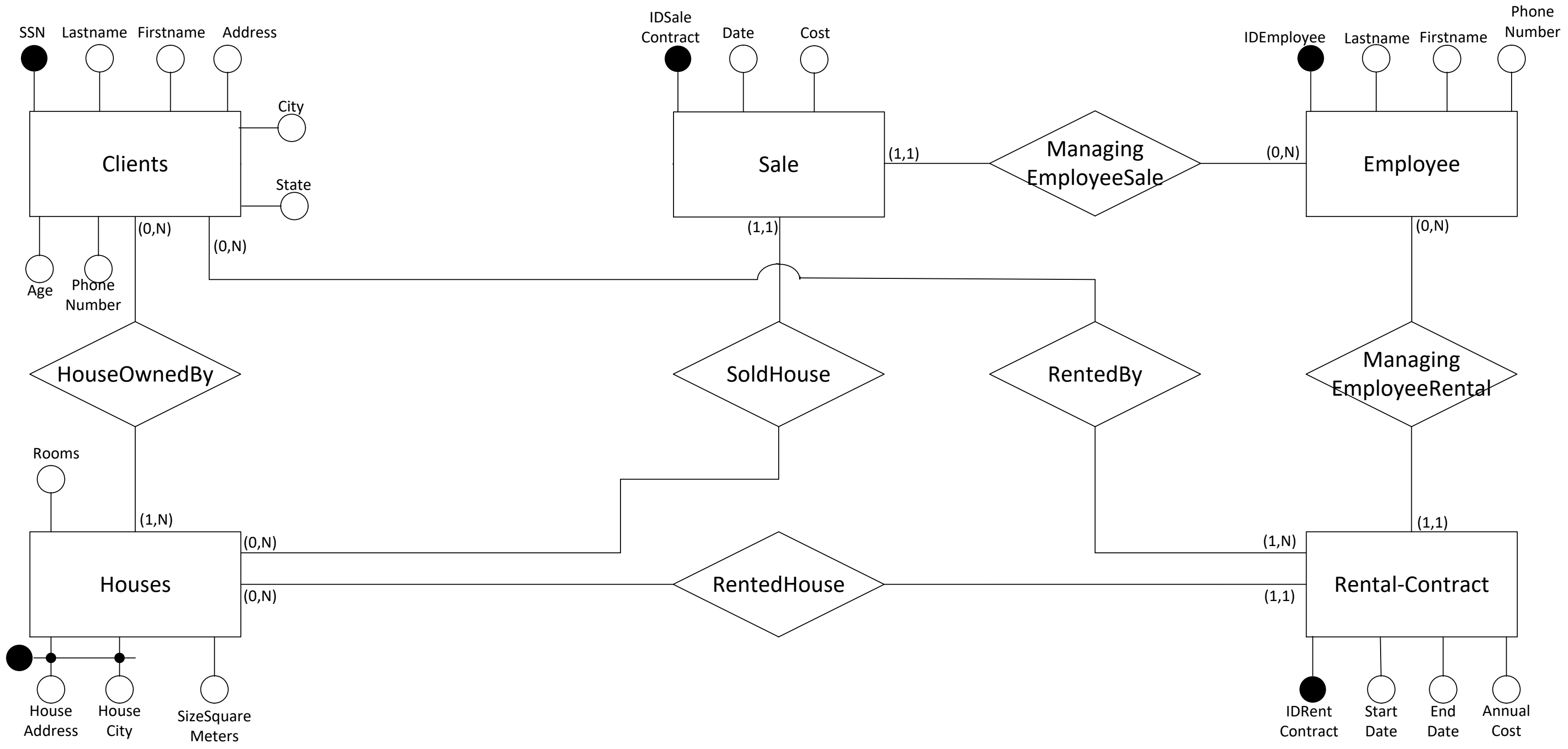
LALuxuryHouses

RENTEDBY (<u>IDRentContract</u>, <u>ClientSSN</u>)



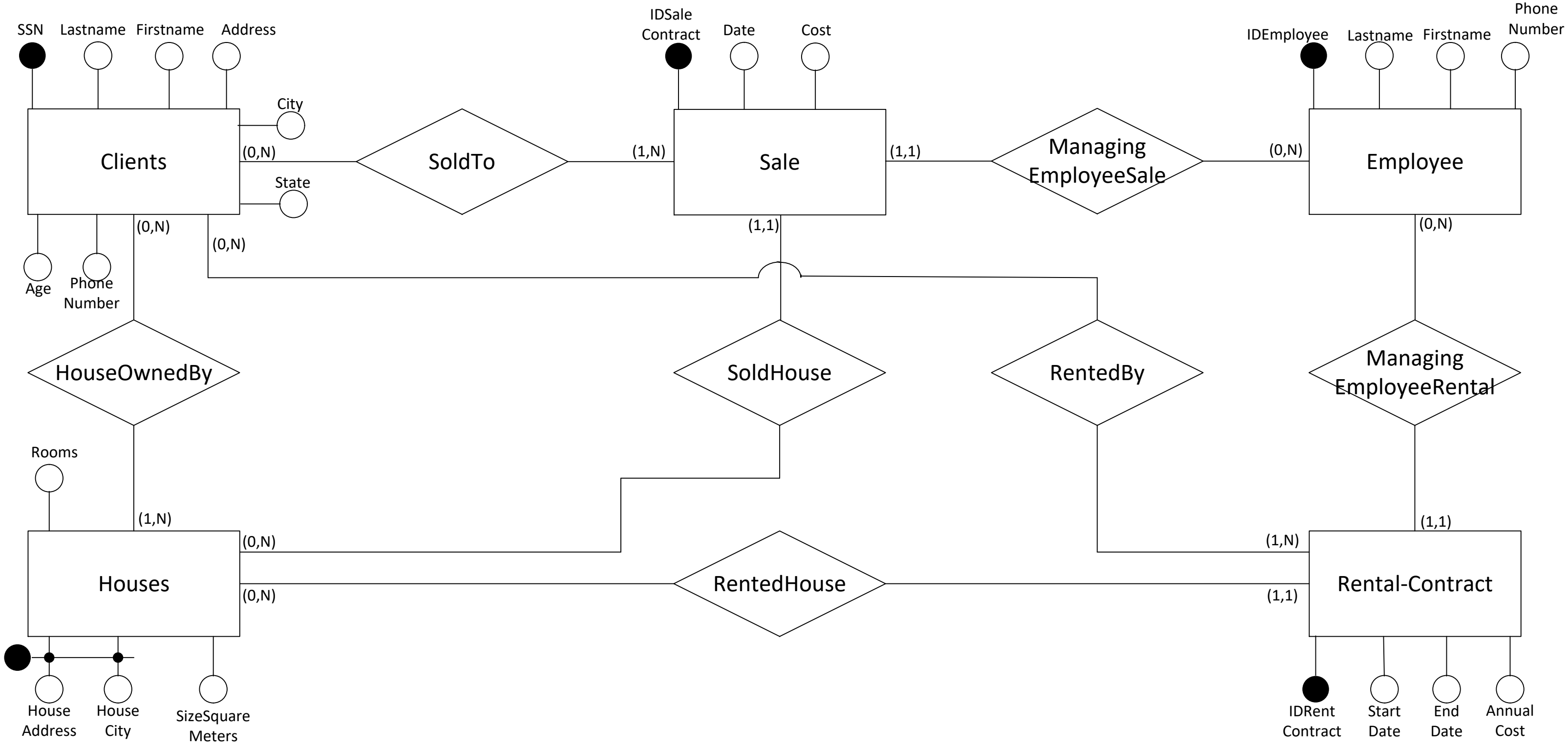
LALuxuryHouses

SALE (IDSaleContract, HouseAddress, HouseCity, Date, Cost, IDEmployee)

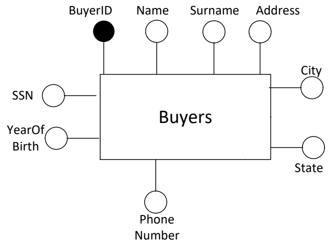


LALuxuryHouses

SOLDTO (IDSaleContract, ClientSSN)

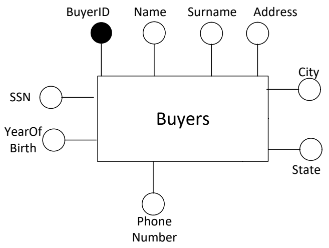


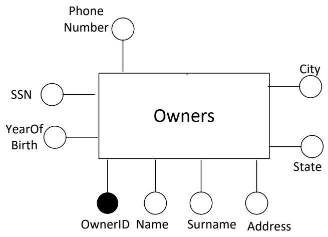
USAHouses



BUYERS (BuyerID, Name, Surname, Address, City, State, YearOfBirth, SSN, PhoneNumber)

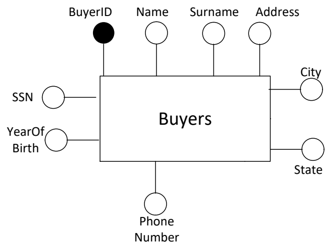
USAHouses

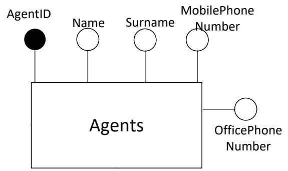


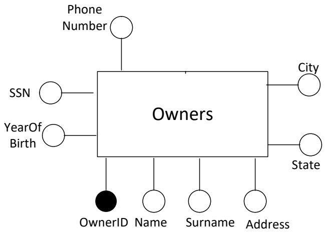


OWNERS (OwnerID, Name, Surname, Address, City, State, YearOfBirth, SSN, PhoneNumber)

USAHouses

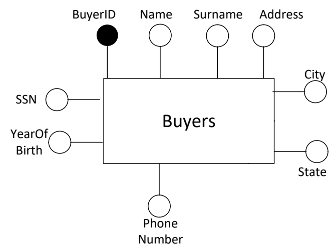


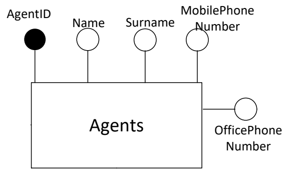


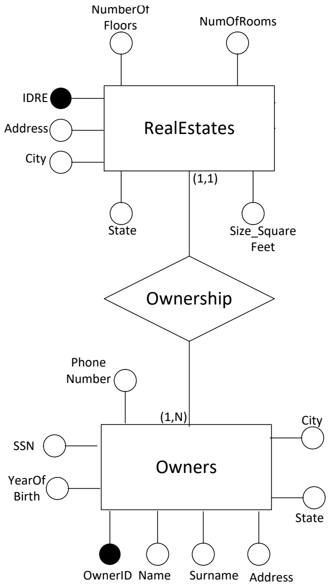


AGENTS (AgentID, Name, Surname, MobilePhoneNumber, OfficePhoneNumber)

USAHouses

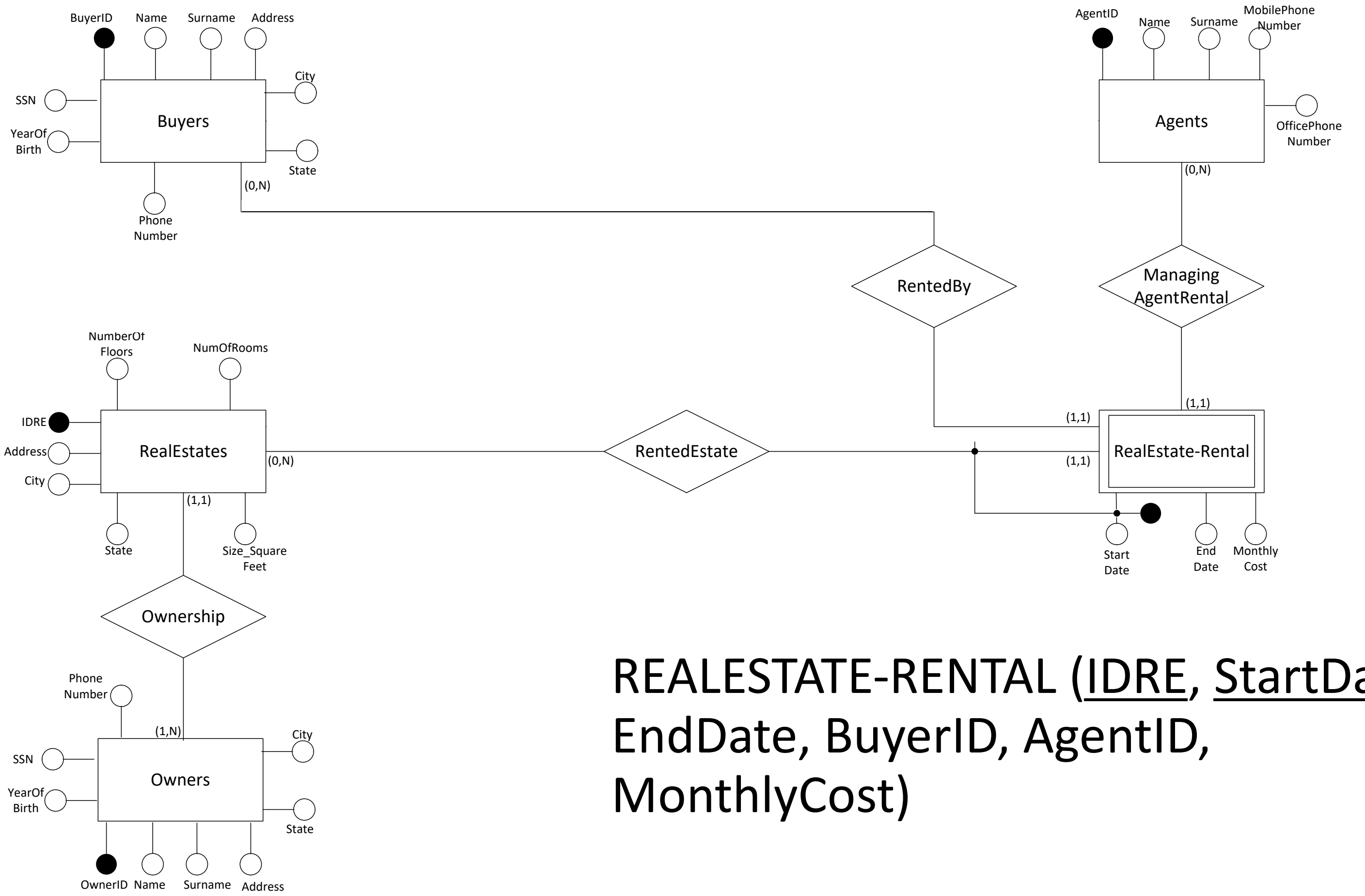






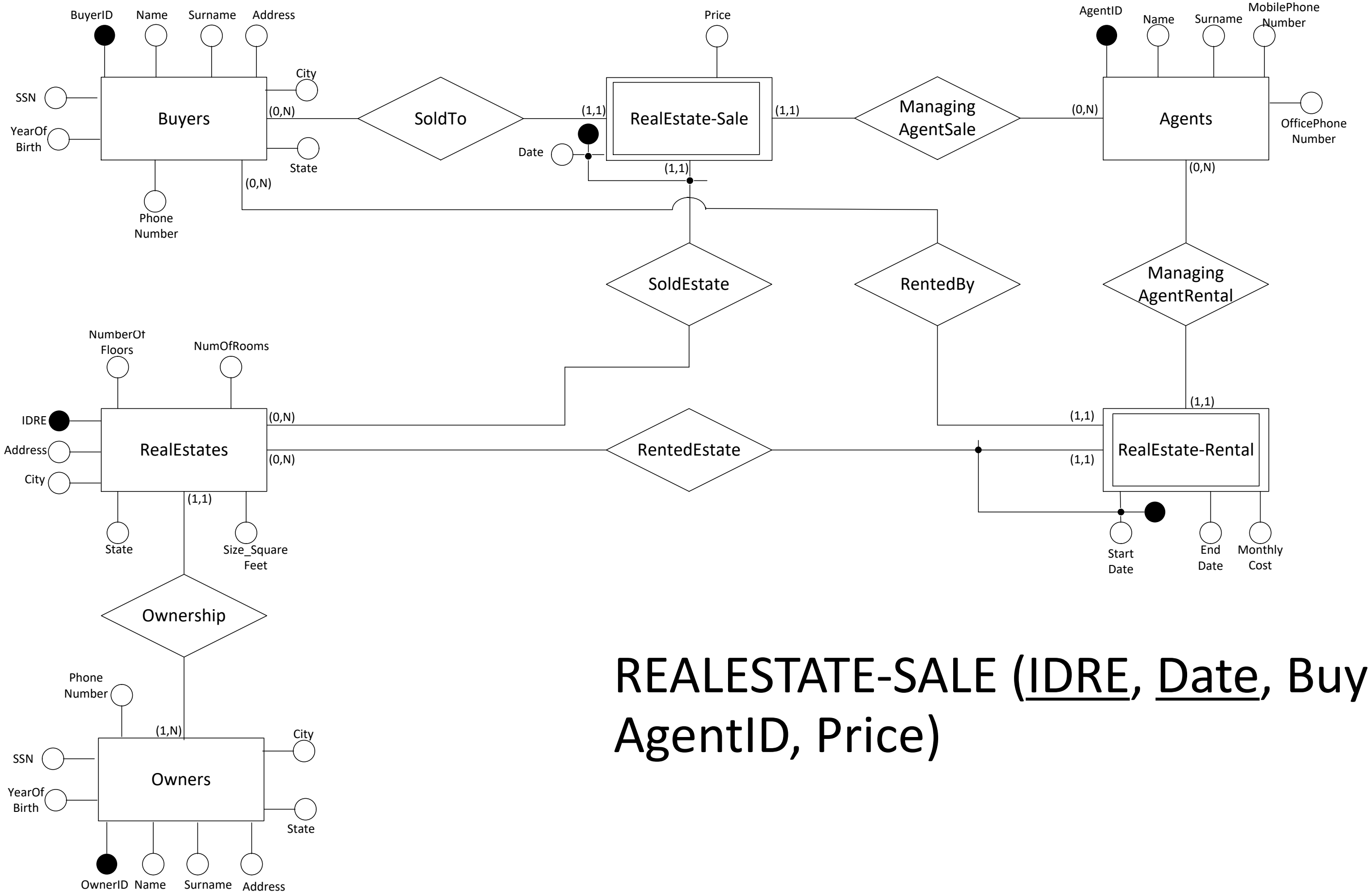
REALESTATES (IDRE, Address, City, State, NumOfRooms, Size_SquareFeet, NumberOfFloors, OwnerID)

USAHouses



REALESTATE-RENTAL (IDRE, StartDate, EndDate, BuyerID, AgentID, MonthlyCost)

USAHouses



REALESTATE-SALE (IDRE, Date, BuyerID, AgentID, Price)

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners		

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients
		- Lastname → Surname	Lastname

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients
		- Lastname → Surname	Lastname
		- Firstname → Name	Firstname

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients
		- Lastname → Surname	Lastname
		- Firstname → Name	Firstname
		Data semantics conflicts	

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients
		- Lastname → Surname	Lastname
		- Firstname → Name	Firstname
		Data semantics conflicts	
		- Age → YearOfBirth	Compute the year of birth from the age

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients
		- Lastname → Surname	Lastname
		- Firstname → Name	Firstname
		Data semantics conflicts	
		- Age → YearOfBirth	Compute the year of birth from the age
		Key conflict	

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Clients	Buyers / Owners	Name conflicts	
		- Entity name	Clients
		- Lastname → Surname	Lastname
		- Firstname → Name	Firstname
		Data semantics conflicts	
		- Age → YearOfBirth	Compute the year of birth from the age
		Key conflict	
		- SSN → BuyerID/OwnerID	Use the SSN, available also in USAHouses

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Houses	RealEstates	Name conflicts	
		- Entity name	RealEstates
		- HouseAddress → Address	Address
		- HouseCity → City	City
		- Rooms → NumOfRooms	Rooms
		Data semantics conflicts	
		- SizeSquareMeters → Size_SquareFeet	Convert in square meters
		Key conflict	
		- HouseAddress+HouseCity → IDRE	Use Address+City, available also in USAHouses
		Cardinality conflicts	
		- More owners → one owner	More owners

A comment on NumberOfFloors: it is present only in USAHouses, so there are no conflicts about it. It will appear in the global schema, but it will be optional because it is not known for the houses in LALuxuryHouses.

Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Employee	Agents	Name conflicts	
		- Entity name	Employee
		- EmployeeID → AgentID	EmployeeID
		- Firstname → Name	Firstname
		- Lastname → Surname	Lastname
		- PhoneNumber → OfficePhoneNumber	OfficePhoneNumber

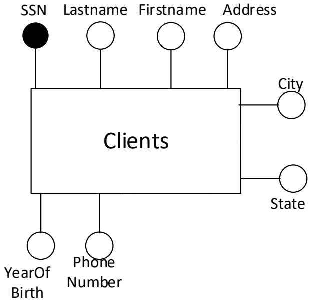
Related concept identification + Conflict analysis and resolution

LALuxuryHouses	USAHouses	Conflicts	Solution
Sale	RealEstate-Sale	Name conflicts	
		- Entity name	Sale
		- Cost → Price	Cost
		Key conflict	
		- IDSaleContract → Date+IDRE	Use IDSaleContract. For USAHouses we obtain the ID concatenating IDRE and Date
		Cardinality conflicts	
		- More buyers → One buyer	More buyers

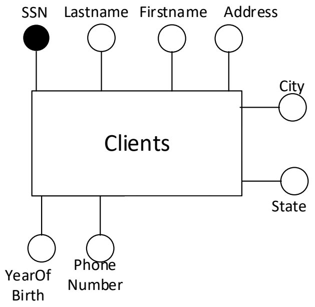
Related concept identification + Conflict analysis and resolution

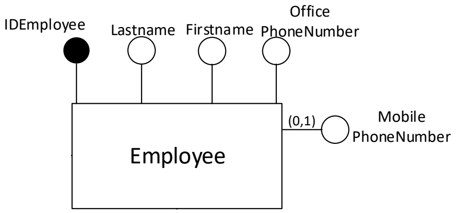
LALuxuryHouses	USAHouses	Conflicts	Solution
Rental-Contract	RealEstate-Rental	Name conflicts	
		- Entity name	Rental-Contract
		Data semantics conflicts	
		- AnnualCost → MonthlyCost	AnnualCost (=MontlyCost*12)
		Key conflict	
		- IDRentContract → IDRE+StartDate	Use IDRentContract. In USAHouses we obtain the ID concatenating IDRE and StartDate
		Cardinality conflicts	
		- More renters → One renter	More renters

Global conceptual schema design

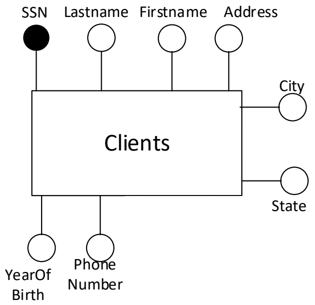


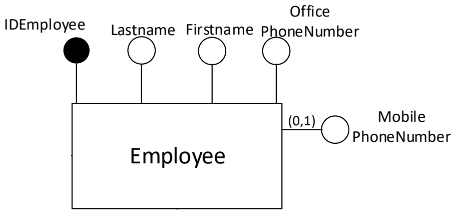
Global conceptual schema design

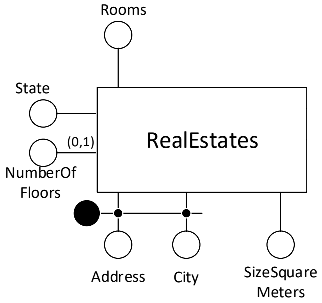




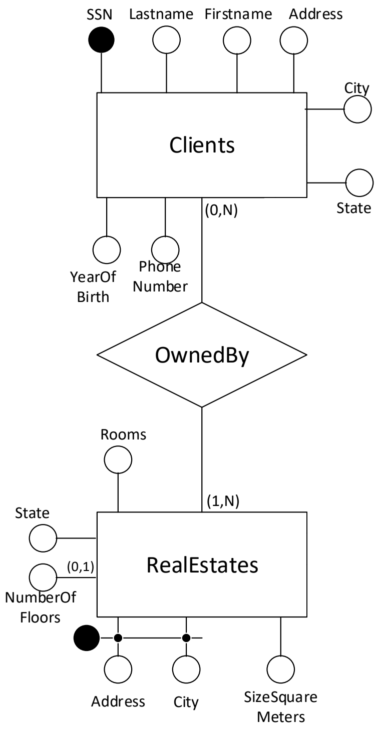
Global conceptual schema design

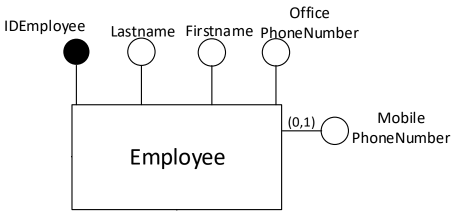




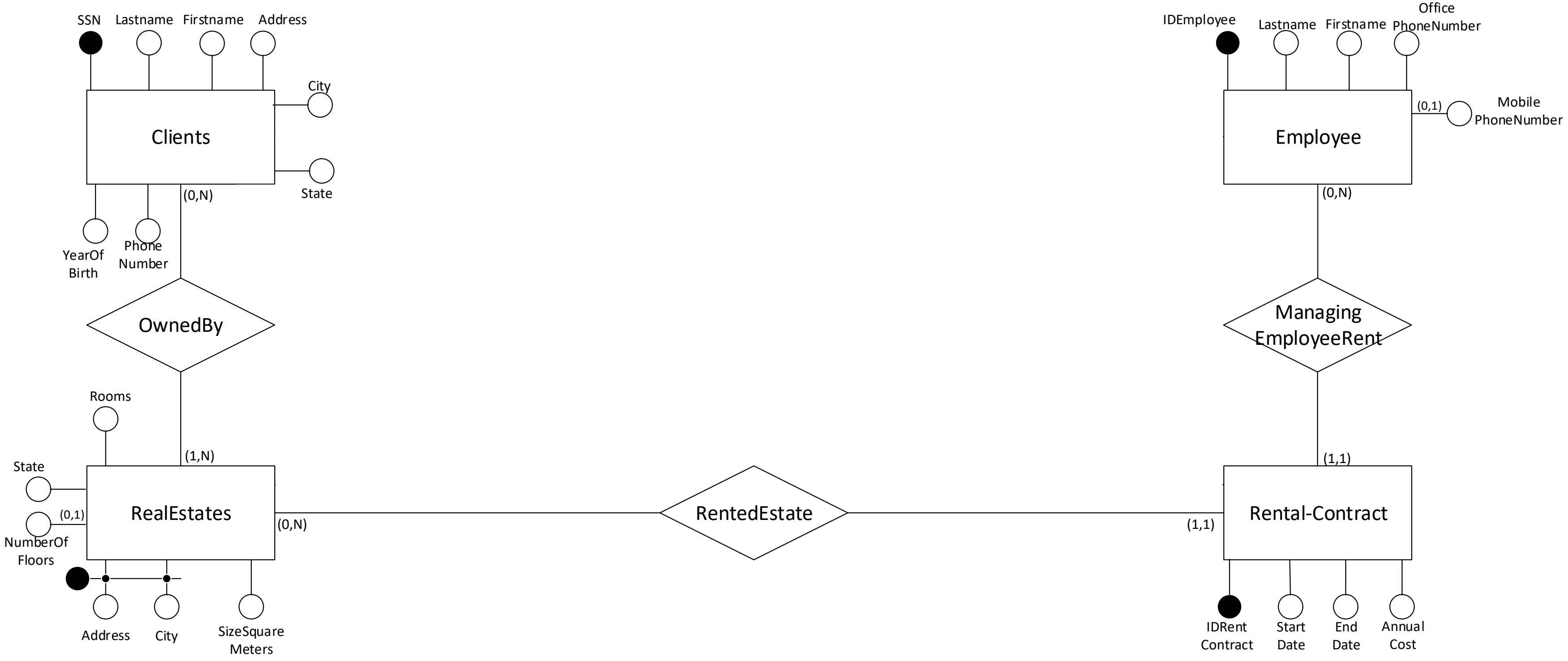


Global conceptual schema design

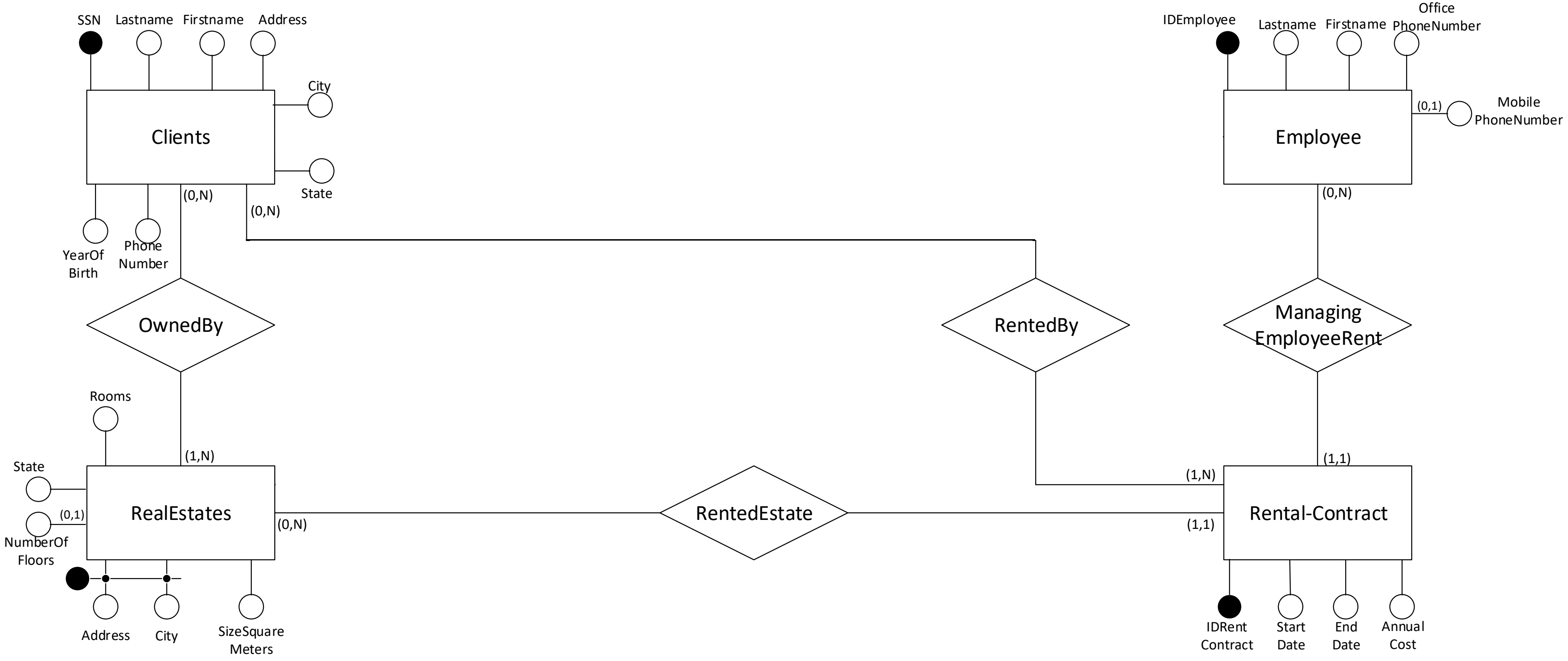




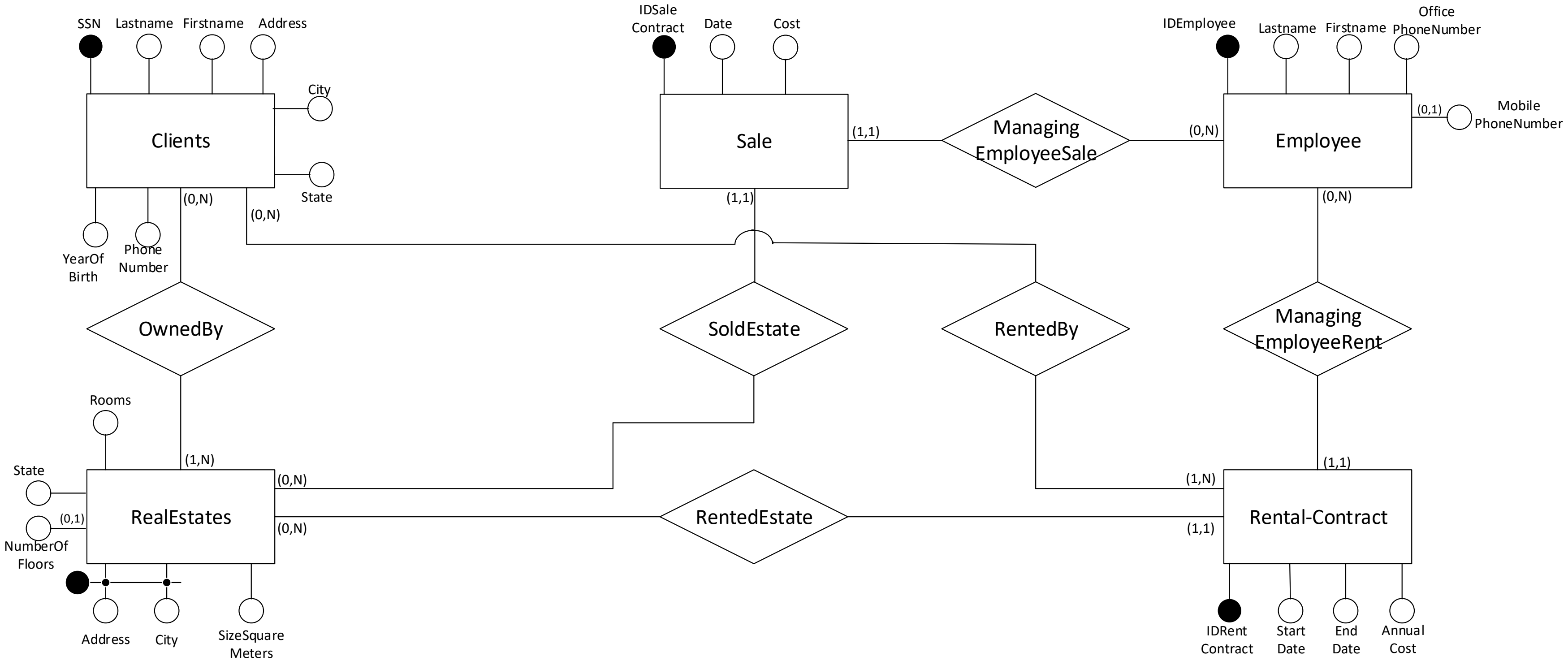
Global conceptual schema design



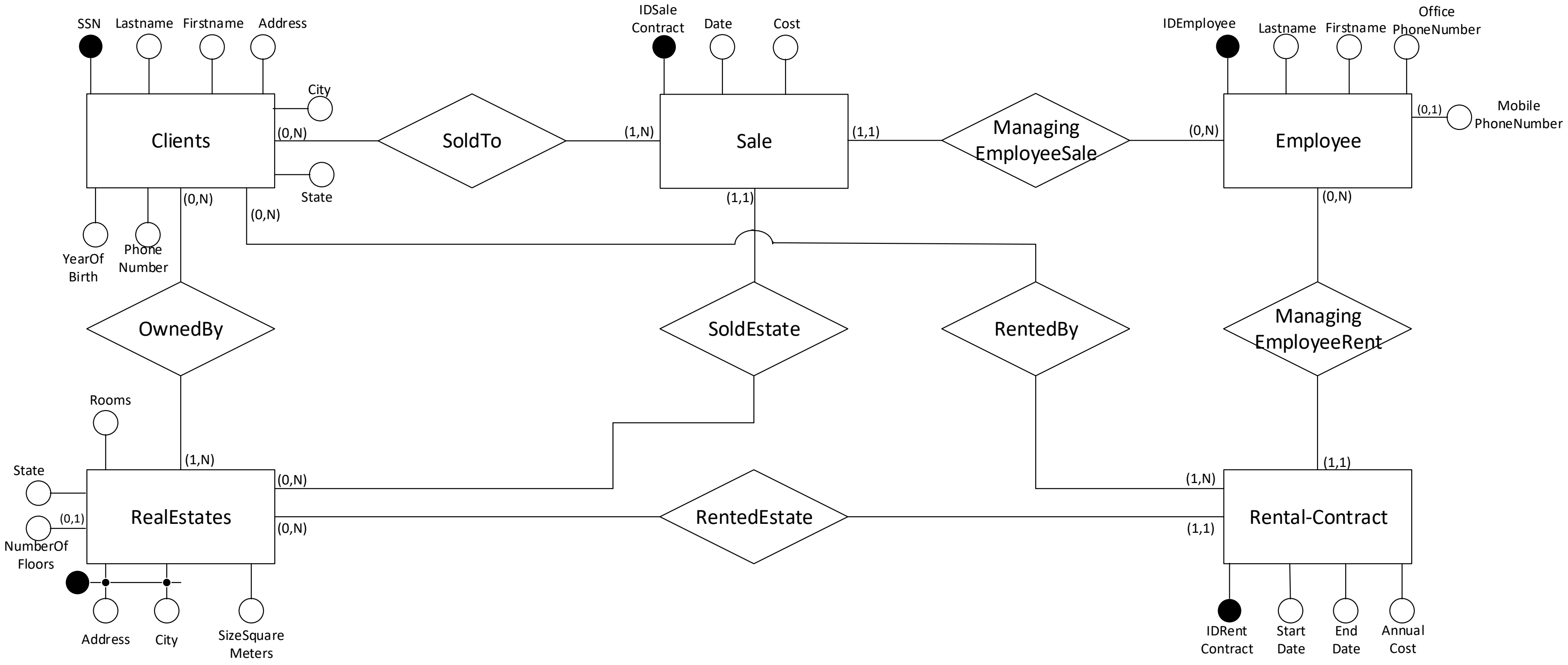
Global conceptual schema design



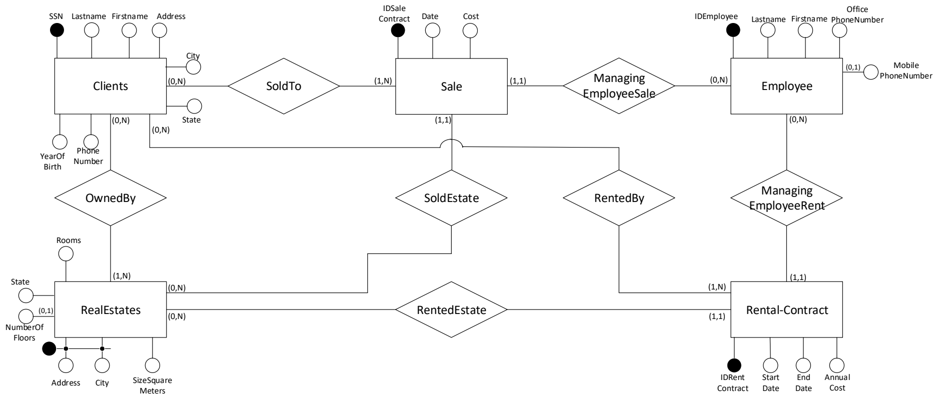
Global conceptual schema design



Global conceptual schema design

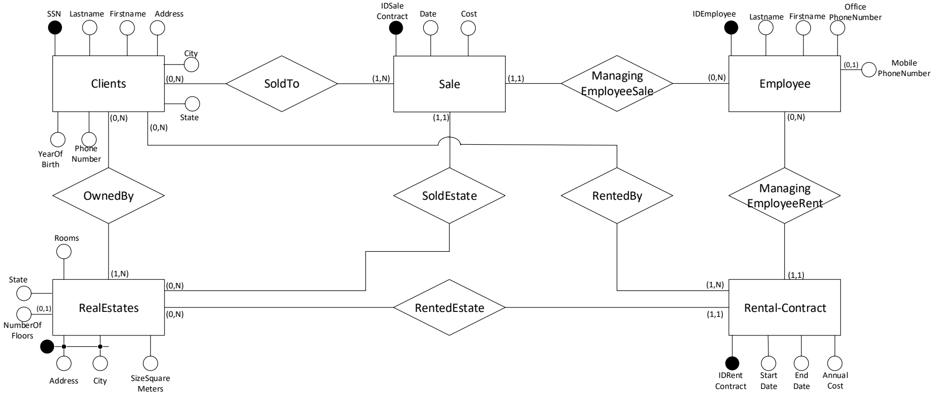


Conceptual to logical translation of the global schema



Clients (<u>SSN</u>, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

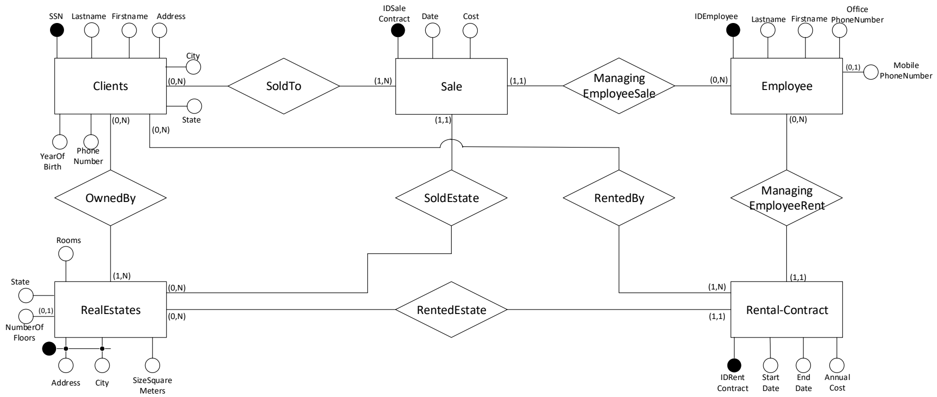
Conceptual to logical translation of the global schema



Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

Conceptual to logical translation of the global schema

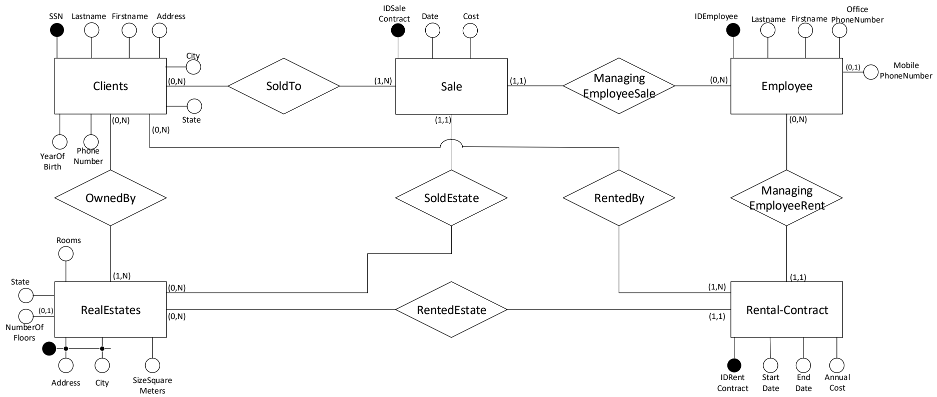


Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

SoldTo (IDSaleContract, Client)

Conceptual to logical translation of the global schema



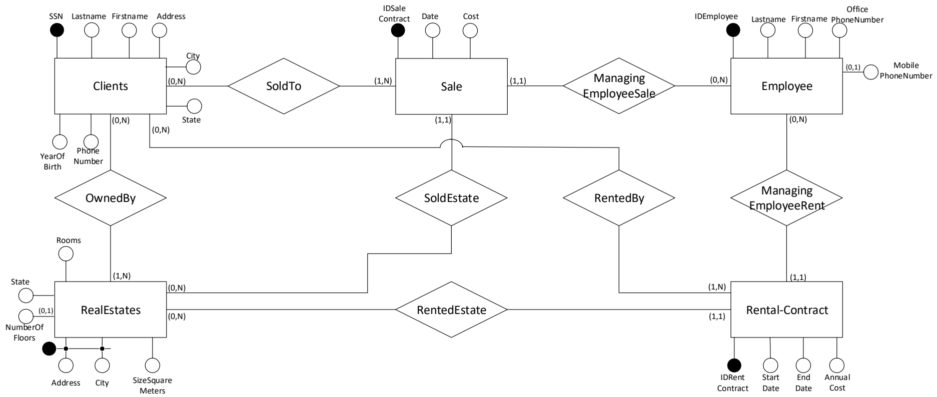
Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

SoldTo (IDSaleContract, Client)

RealEstates (Address, City, State, SizeSquareMeters, Rooms, NumberOfFloors*)

Conceptual to logical translation of the global schema



Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

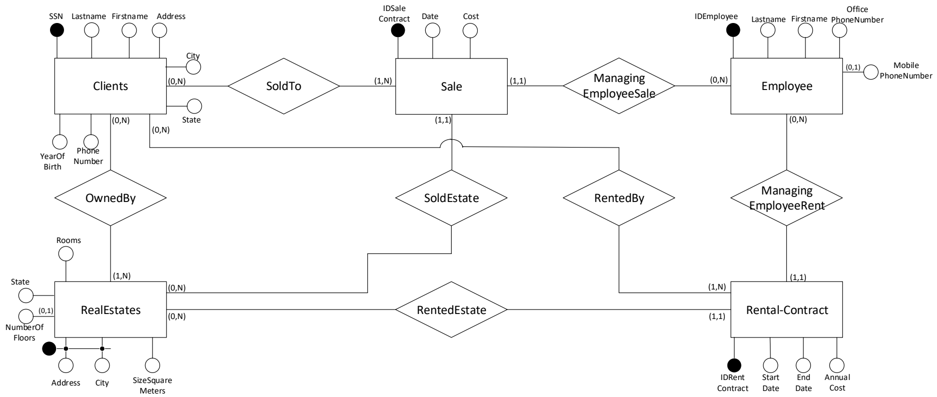
Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

SoldTo (IDSaleContract, Client)

RealEstates (Address, City, State, SizeSquareMeters, Rooms, NumberOfFloors*)

Rental-Contract (IDRentalContract, StartDate, EndDate, AnnualCost, IDEmployee, EstateAddress, EstateCity)

Conceptual to logical translation of the global schema



Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

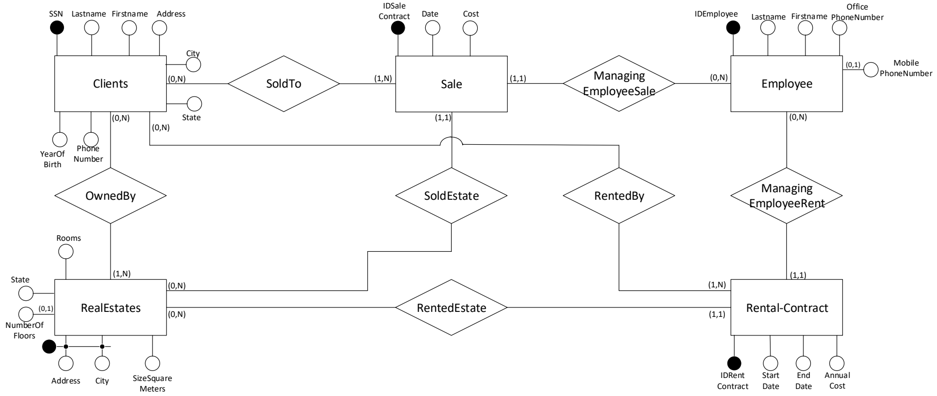
SoldTo (IDSaleContract, Client)

RealEstates (Address, City, State, SizeSquareMeters, Rooms, NumberOfFloors*)

Rental-Contract (IDRentalContract, StartDate, EndDate, AnnualCost, IDEmployee, EstateAddress, EstateCity)

RentedBy (IDRentalContract, Client)

Conceptual to logical translation of the global schema



Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

SoldTo (IDSaleContract, Client)

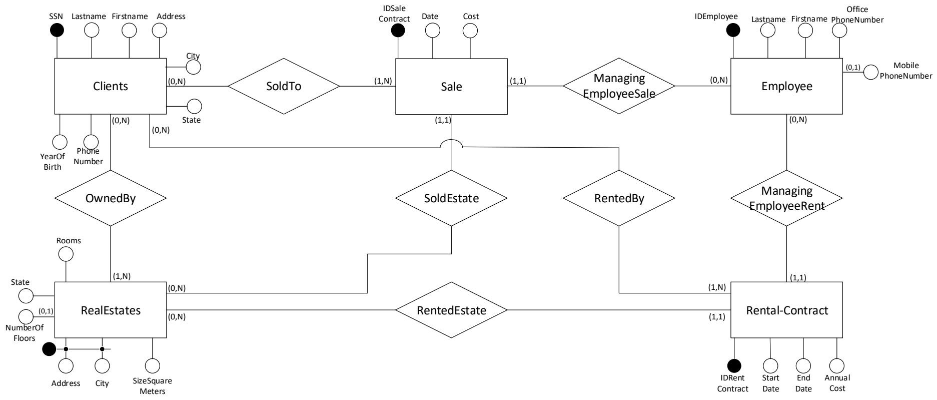
RealEstates (Address, City, State, SizeSquareMeters, Rooms, NumberOfFloors*)

Rental-Contract (IDRentalContract, StartDate, EndDate, AnnualCost, IDEmployee, EstateAddress, EstateCity)

RentedBy (IDRentalContract, Client)

OwnedBy (EstateAddress, EstateCity, Client)

Conceptual to logical translation of the global schema



Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth, PhoneNumber)

Sale (IDSaleContract, Date, Cost, EstateAddress, EstateCity, IDEmployee)

SoldTo (IDSaleContract, Client)

RealEstates (Address, City, State, SizeSquareMeters, Rooms, NumberOfFloors*)

Rental-Contract (IDRentalContract, StartDate, EndDate, AnnualCost, IDEmployee, EstateAddress, EstateCity)

RentedBy (IDRentalContract, Client)

OwnedBy (EstateAddress, EstateCity, Client)

Employee (IDEmployee, Lastname, Firstname, OfficePhoneNumber, MobilePhoneNumber*)

GAV mapping

```
CREATE VIEW GlobalSchema.TableX(attributes of GlobalSchema.TableX
in the order as in GlobalSchema.TableX) AS (

          SELECT      attributes from SourceSchema1 but in the order as
                      in GlobalSchema.TableX
SOURCE 1  FROM        SourceSchema1.TableY1, SourceSchema1.TableY2, …
          WHERE       …

          UNION

          SELECT      attributes from SourceSchema2 but in the order as
                      in GlobalSchema.TableX
SOURCE 2  FROM        SourceSchema2.TableZ1, SourceSchema2.TableZ2, …
          WHERE       …
)
```

GAV mapping - Clients

```
CREATE VIEW USARealEstateCompany.Clients (SSN, Lastname, Firstname, Address, City, State, YearOfBirth,
PhoneNumber) AS (
          SELECT SSN, Lastname, Firstname, Address, City, State, CurrentYear()-Age, PhoneNumber
          FROM LALuxuryHouses.Clients

          UNION

          SELECT SSN, Surname, Name, Address, City, State, YearOfBirth, PhoneNumber
          FROM USAHouses.Buyers

          UNION

          SELECT SSN, Surname, Name, Address, City, State, YearOfBirth, PhoneNumber
          FROM USAHouses.Owners
)
```

GAV mapping - RealEstates

```
CREATE VIEW USARealEstateCompany.RealEstates (Address, City, State,
SizeSquareMeters, Rooms, NumberOfFloors) AS (
        SELECT HouseAddress, HouseCity, ‘California’, SizeSquareMeters,
                Rooms, null
        FROM LALuxuryHouses.Houses


        UNION


        SELECT Address, City, State, SizeSquareFeet*0.0929, NumOfRooms,
                NumberOfFloors
        FROM USAHouses.RealEstates
)
```

Keygen

Since we do data integration with the union operator we are assuming everything to be disjoint, but…

… it does not mean primary keys are unique!

Source1	**Source2**
ID=1 Name=”A”	ID=1 Name=”F”
ID=2 Name=”B”	ID=2 Name=”G”
ID=3 Name=”C”	ID=3 Name=”H”
ID=4 Name=”D”	ID=4 Name=”I”
ID=5 Name=”E”	

GAV mapping - Sale

```
CREATE VIEW USARealEstateCompany.Sale (IDSaleContract, Date, Cost, EstateAddress,
EstateCity,
IDEmployee) AS (                   123410122019-LALuxuryHouses
        SELECT KeyGenSale (IDSaleContract, ‘LALuxuryHouses’), Date, Cost, HouseAddress,
                HouseCity, KeyGenEmployee(IDEmployee, ‘LALuxuryHouses’)
        FROM LALuxuryHouses.Sale

        UNION
                                1234       10122019
        SELECT KeyGenSale(R.IDRE||S.Date, ‘USAHouses’), S.Date, S.Price, R.Address, R.City,
                KeyGenEmployee(S.AgentID, ‘USAHouses’)
        FROM USAHouses.RealEstate-Sale AS S, USAHouses.RealEstates AS R
        WHERE S.IDRE=R.IDRE
)
```

GAV mapping - SoldTo

```
CREATE VIEW USARealEstateCompany.SoldTo (IDSaleContract, Client) AS (
        SELECT KeyGenSale(IDSaleContract, ‘LALuxuryHouses’), ClientSSN
        FROM LALuxuryHouses.SoldTo

        UNION

        SELECT KeyGenSale(R.IDRE||R.Date, ‘USAHouses’), B.SSN
        FROM USAHouses.RealEstate-Sale AS R, USAHouses.Buyers AS B
        WHERE R.BuyerID = B.BuyerID
)
```

GAV mapping - OwnedBy

```
CREATE VIEW USARealEstateCompany.OwnedBy (EstateAddress, EstateCity,
Client) AS (
        SELECT HouseAddress, HouseCity, ClientSSN
        FROM LALuxuryHouses.House-OwnedBy

        UNION

        SELECT R.Address, R.City, O.SSN
        FROM RealEstates AS R, Owners AS O
        WHERE R.OwnerID=O.OwnerID
)
```

GAV mapping - Employee

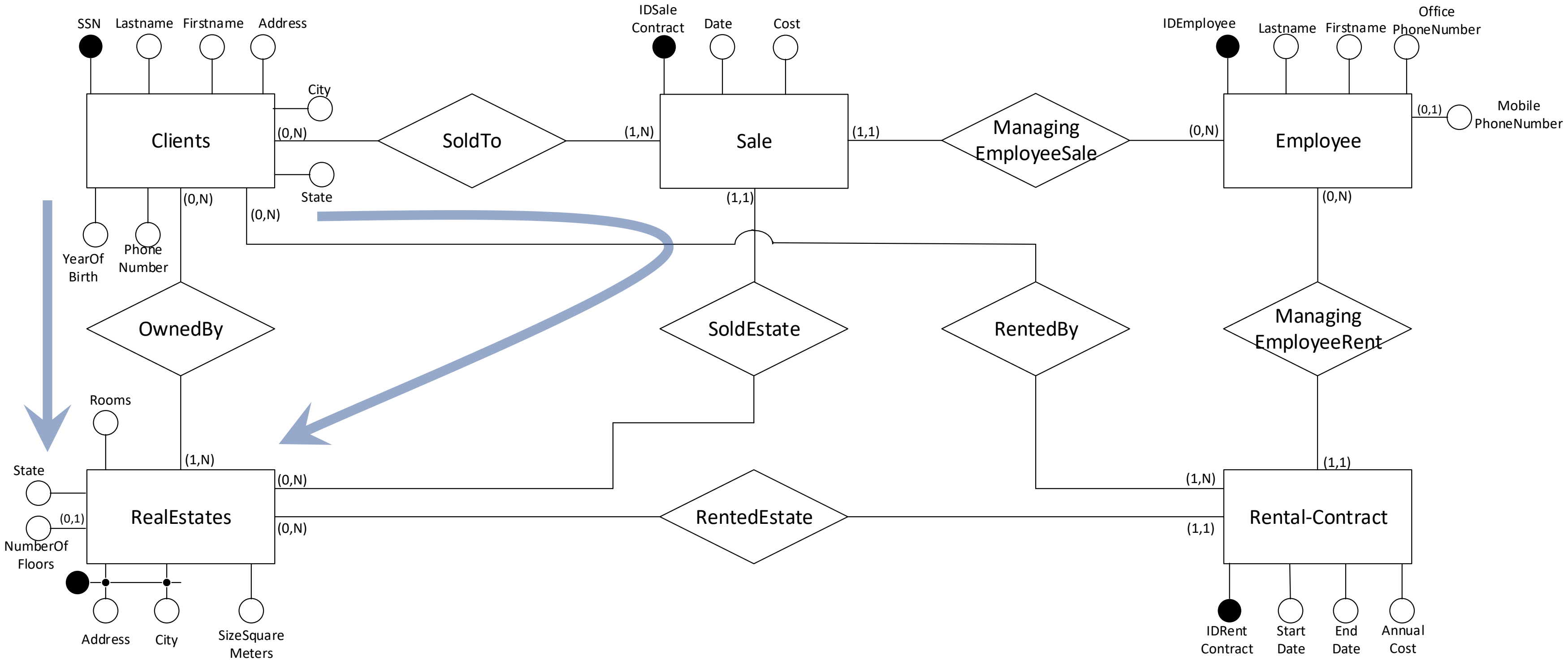
```
CREATE VIEW USARealEstateCompany.Employee (IDEmployee, Lastname,
Firstname, OfficePhoneNumber, MobilePhoneNumber) AS (
        SELECT KeyGenEmployee(IDEmployee, ‘LALuxuryHouses’), Lastname,
                Firstname, PhoneNumber, null
        FROM LALuxuryHouses.Employee

        UNION

        SELECT KeyGenEmployee(AgentID, ‘USAHouses’), Surname, Name,
                OfficePhoneNumber, MobilePhoneNumber
        FROM USAHouses.Agents
)
```

Query formulation on the global schema

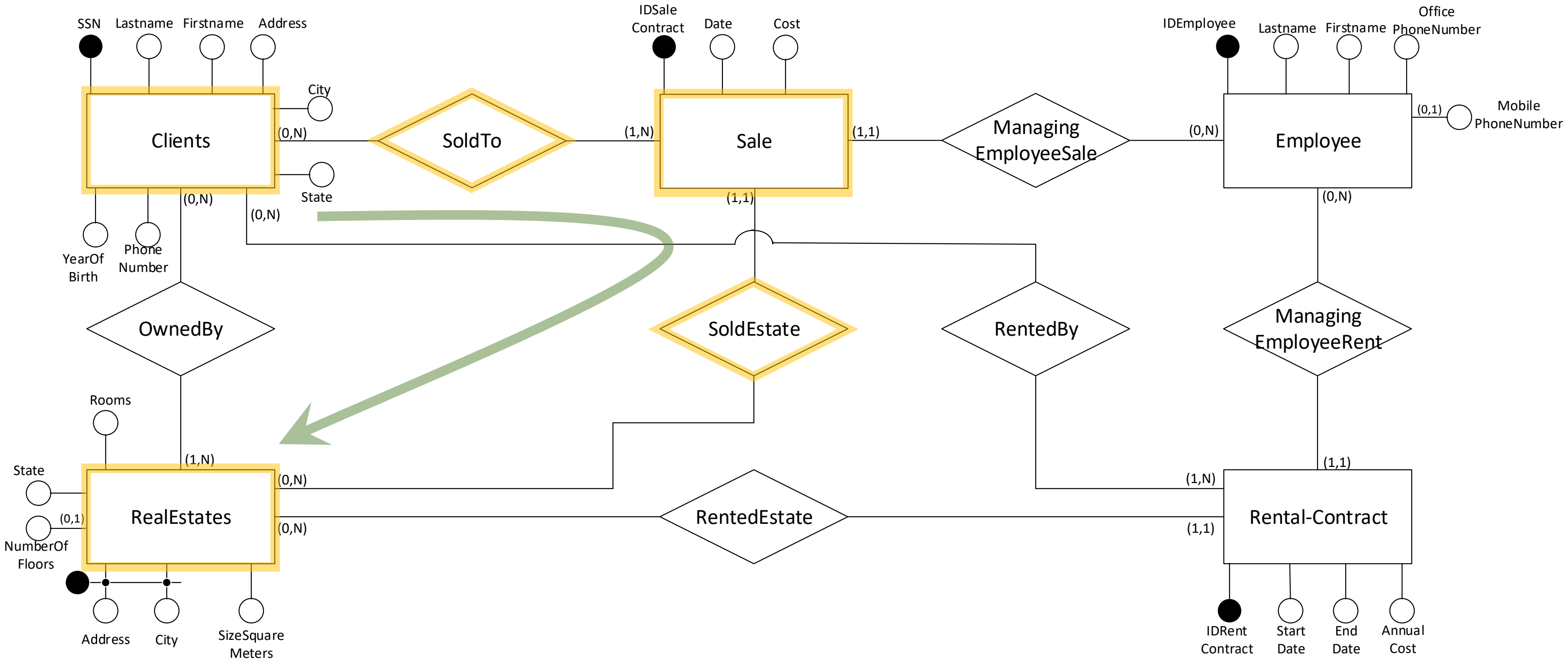
Consider query Q posed on USARealEstateCompany’s schema and write it either in Datalog or SQL.

Q: *“Find the name and surname of the buyers who live in the city of Los Angeles and have bought at least one house larger than 100 square meters located in the city of Beverly Hills”.*

Global conceptual schema design



Global conceptual schema design



Query formulation on the global schema

Consider query Q posed on USARealEstateCompany's schema and write it either in Datalog or SQL.

Q: *"Find the name and surname of the buyers who live in the city of Los Angeles and have bought at least one house larger than 100 square meters located in the city of Beverly Hills".*

```
SELECT DISTINCT C.Lastname, C.Firstname
FROM USARealEstateCompany.Clients AS C, USARealEstateCompany.Sale AS S,
        USARealEstateCompany.SoldTo AS ST, USARealEstateCompany.RealEstates
AS R
WHERE C.SSN = ST.Client AND ST.IDSaleContract=S.IDSaleContract AND
        S.EstateAddress=R.Address AND S.EstateCity=R.City AND
        R.SizeSquareMeters>100 AND C.City='Los Angeles' AND R.City='Beverly
        Hills'
```

Query rewriting

```
SELECT C.Lastname, C.Firstname
FROM LALuxuryHouses.Clients AS C, LALuxuryHouses.SoldTo AS ST,
        LALuxuryHouses.Sale AS S, LALuxuryHouses.Houses AS H
WHERE C.SSN=ST.ClientSSN AND ST.IDSaleContract=S.IDSaleContract AND
        H.HouseAddress=S.HouseAddress AND H.HouseCity=S.HouseCity AND
        C.City=’Los Angeles’ AND H.SizeSquareMeters>100 AND H.City=’Beverly
        Hills’

UNION

SELECT B.Surname AS Lastname, B.Name AS Firstname
FROM USAHouses.Buyers AS B, USAHouses.RealEstate-Sale AS S,
        USAHouses.RealEstates AS R
WHERE B.BuyerID=S.BuyerID AND S.IDRE=R.IDRE AND B.City =’Los Angeles’ AND
        R.SizeSquareFeet*0.0929>100 AND R.City=’Beverly Hills’
```