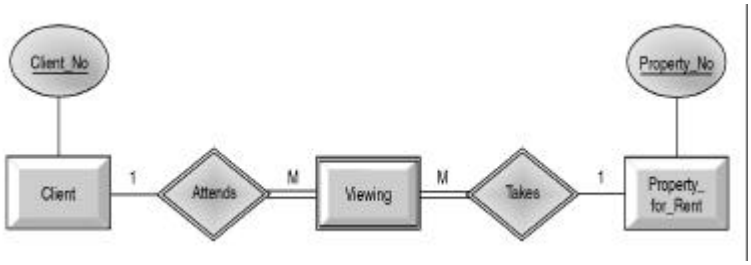
Chapter 11

Logical Databases Design Methodology
Worked Example
Transparencies

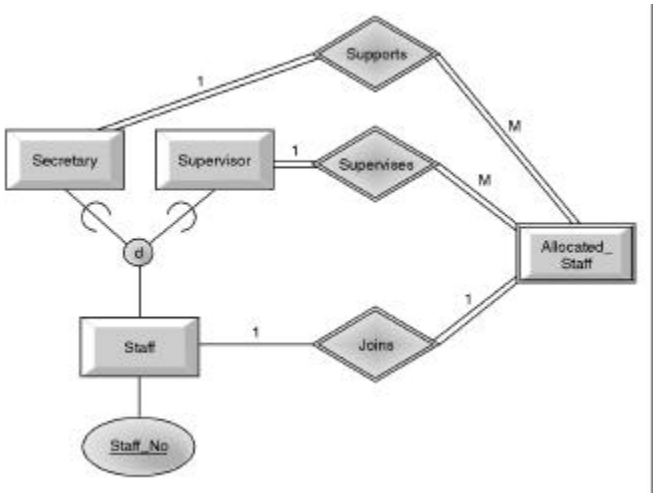
Chapter 11 - Objectives

- **How to use the logical database design methodology, described in Chapter 8.**
- **How to use this methodology to create a logical database design for the *DreamHome* case study.**

Removing Client *Views* Property_for_Rent (M:N) relationship



Removing recursive *Supervises* and *Supports* relationships

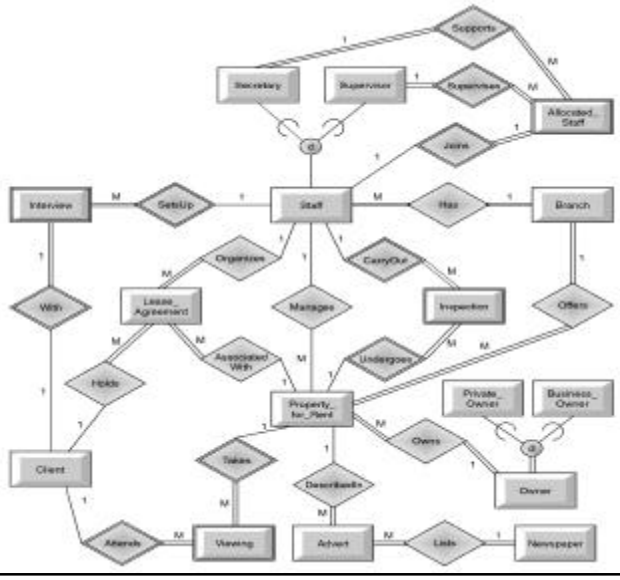


Step 2.2 Derive relations from local logical data model

- **To represent entities and relationships described in Supervisor's view of *DreamHome.***
- **For example, the composition of Viewing relation is**

Viewing (Property_No, Client_No, Date_View, Comments)
Primary Key Property_No, Client_No, Date_View
Foreign Key Property_No **references** Property_for_Rent(Property_No)
Foreign Key Client_No **references** Client(Client_No)

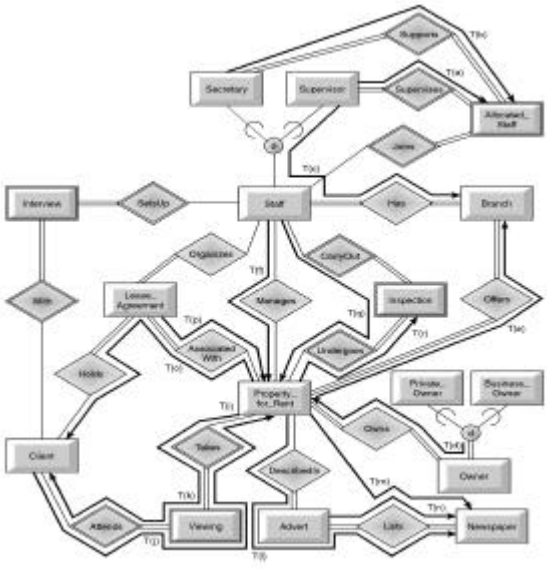
Supervisor's local logical data model (Version 1)



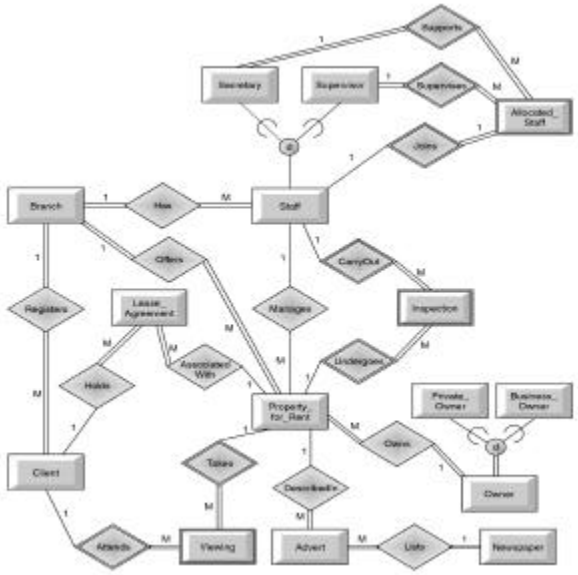
Step 2.4 Validate model against user transactions

- **We can ensure that the information (entities, relationships, and attributes) required by each transaction is supported by the model by providing a description of how we may achieve the transaction.**
- **We can diagrammatically representing each transaction on the Supervisor's local logical data model.**

Supervisor's local logical data model displaying the transactions supported



Step 2.5 Draw Supervisor's local logical data model (Final Version)



Step 2.6 Define integrity constraints

- Required data
- Attribute domain constraints
- Entity integrity
- Referential integrity
- Enterprise constraints

Comparison of entities and their primary keys in Supervisor's and Manager's views

Entity type (Supervisor's view)	*Primary key*	*Entity type (Manager's view)*	*Primary key*
Branch	**Branch_No**	**Branch**	**Branch_No**
Staff	**Staff_No**	**Staff**	**Staff_No**
Supervisor	**Staff_No**	**Supervisor**	**Staff_No**
Secretary	Staff_No		
Allocated_Staff	**Staff_No**	**Allocated_Staff**	**Staff_No**
		Manager	Staff_No
		Next_of_Kin	Staff_No, NName
Property_for_Rent	**Property_No**	**Property_for_Rent**	**Property_No**
Private_Owner	**Owner_No**	**Private_Owner**	**Owner_No**
Business_Owner	**Owner_No**	**Business_Owner**	**Owner_No**
Advert	**Advert_No**	**Advert**	**Property_No, Date_Advert, Newspaper_Name**
Newspaper	**Newspaper_Name**	**Newspaper**	**Newspaper_Name**
Client	Client_No		
		Renter	Renter_No
Viewing	**Property_No, Client_No, Date_View**	**Viewing**	**Property_No, Renter_No, Date_View**
Lease_Agreement	Lease_No		
		Rental_Agreement	Rental_No
Inspection	Property_No, Staff_No, Date_Inspect		

Comparison of relationships in Supervisor's and Manager's views

Entity type (Supervisor's view)	*Relationship type*	*Entity type (Supervisor's view)*	*Entity type (Manager's view)*	*Relationship type*	*Entity type (Manager's view)*
Branch	***Has***	**Staff**	**Branch**	***Has***	**Staff**
	Offers	**Property_for_Rent**		***Offers***	**Property_for_Rent**
	Registers	Client		*RefersTo*	Renter
Staff	*Manages*	Property_for_Rent	Staff	*Oversees*	Property_for_Rent
	CarryOut	Inspection		*RelatedTo*	Next_of_Kin
	Joins	**Allocated_Staff**		***Joins***	**Allocated_Staff**
Supervisor	***Supervises***	**Allocated_Staff**	**Supervisor**	***Supervises***	**Allocated_Staff**
Secretary	*Supports*	Allocated_Staff			
			Manager	*Manages*	Branch
Property_for_Rent	*AssociatedWith*	Lease_Agreement	Property_for_Rent	*LinkedTo*	Rental_Agreement
	DescribedIn	Advert		*PlacedIn*	Advert
	Undergoes	Inspection		***Takes***	**Viewing**
	Takes	**Viewing**			
Private_Owner	***Owns***	**Property_for_Rent**	**Private_Owner**	***Owns***	**Property_for_Rent**
Business_Owner	***Owns***	**Property_for_Rent**	**Business_Owner**	***Owns***	**Property_for_Rent**
Newspaper	*Lists*	Advert	Newspaper	*Displays*	Advert
Client	*Attends*	Viewing			
	Holds	Lease_Agreement			
			Renter	*Requests*	Viewing
				Holds	Rental_Agreement

Merging the Staff entities from Supervisor's and Manager's views

(Supervisor's View)

Staff (Staff_No, FName, LName, Address, Tel_No, Sex, DOB (Date_of_Birth), Job_Title, Typing_Speed, Branch_No)
Primary Key Staff_No
Foreign Key Branch_No **references** Branch(Branch_No)

(Manager's View)

Staff (Staff_No, FName, LName, Address, Tel_No, Sex, DOB (Date_of_Birth), Position, **Salary, Date_Joined, NIN (National Insurance Number)**, Typing_Speed, Branch_No)
Primary Key Staff_No
Alternate Key NIN
Foreign Key Branch_No **references** Branch(Branch_No)

(Global View)

Staff (Staff_No, FName, LName, Address, Tel_No, Sex, DOB (Date_of_Birth), Position, Salary, Date_Joined, NIN (National Insurance Number), Typing_Speed, Branch_No)
Primary Key Staff_No
Alternate Key NIN
Foreign Key Branch_No **references** Branch(Branch_No)

Step 3.2 Validate global logical data model

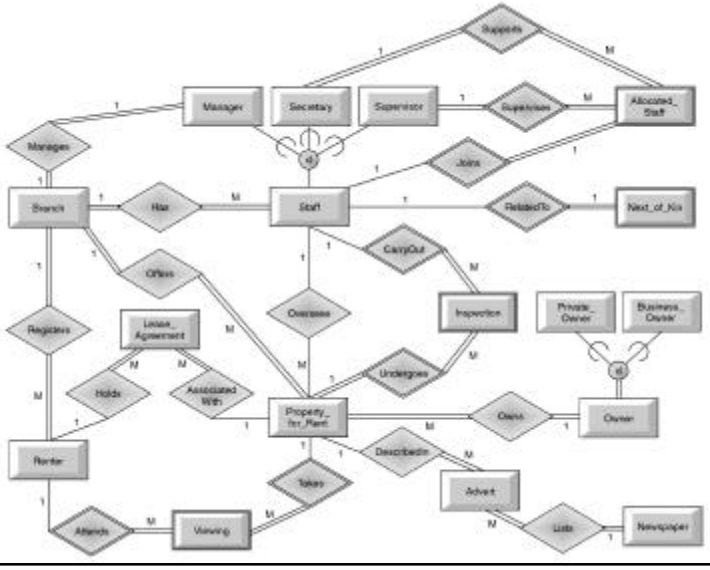
- **Although validated the Supervisor's and Manager's data models before building global logical data model - we may have introduced errors during the process of merging the data models.**

- **Very important to validate the global logical data model using the rules of normalization and against the required transactions.**

Step 3.3 Check for future growth

- **Important global logical data model is capable of being extended at a later stage as the users' requirements change.**

Step 3.4 Draw Global logical data model of *DreamHome* case study



Step 3.5 Review global logical data model with users

- Important to review global logical data model with users of each view. If the model contains any errors we must repeat the appropriate step(s) in the methodology.
- Process of review is repeated until all the users are satisfied with the global logical data model.
- When data model is ‘signed off’ by the users, we proceed to physical design of the database.