Session 9

Database Systems: Design, Implementation, and Management

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

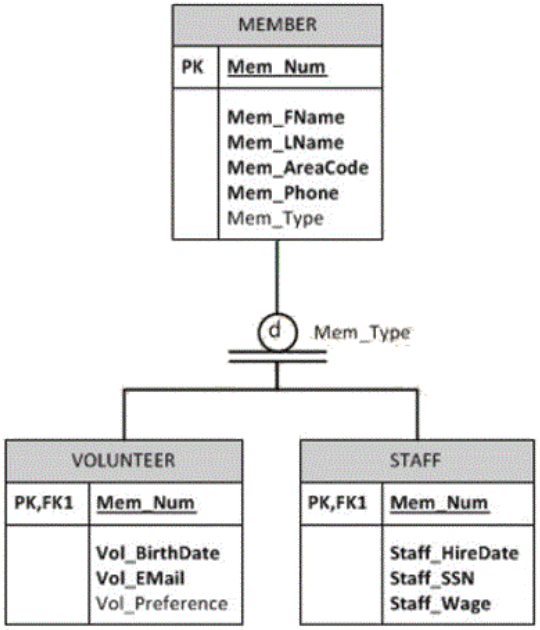
Faculty of Engineering, Built Environment and Information Technology

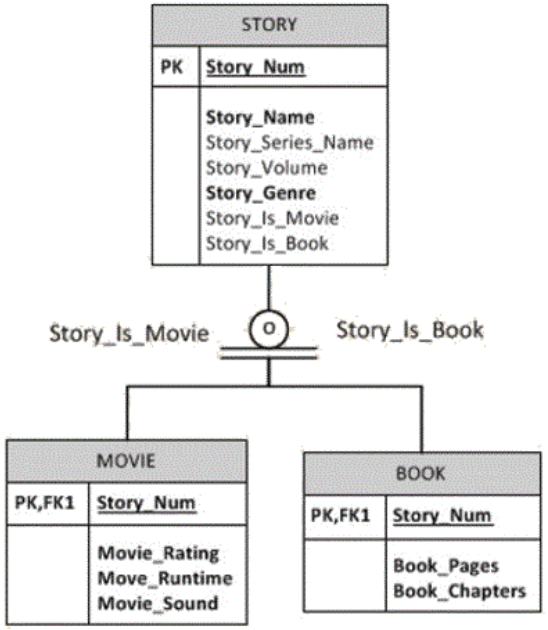
Fakulteit Ingenieurswese, Bou-omgewing en Inligtingtegnologie / Lefapha la Boetšenere, Tikologo ya Kago le Theknolotši ya Tshedimošo

Chapter 5: Advanced Data Modelling

Page 174 (5-1f) to 186 (5-4d)

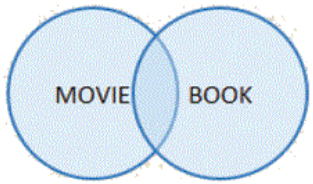
Reminder - Disjoint and Overlap





disjoint

A MEMBER can be in VOLUNTEER or STAFF, but not both

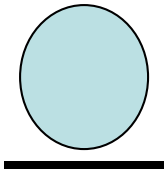


overlapping

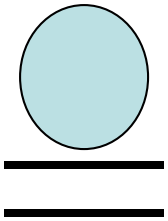
A STORY can be in MOVIE, BOOK, or both

Completeness Constraint

- Specifies whether entity supertype occurrence must be a member of at least one subtype
- **Partial completeness**
  - Symbolized by a circle over a single line
  - Some supertype occurrences are not members of any subtype

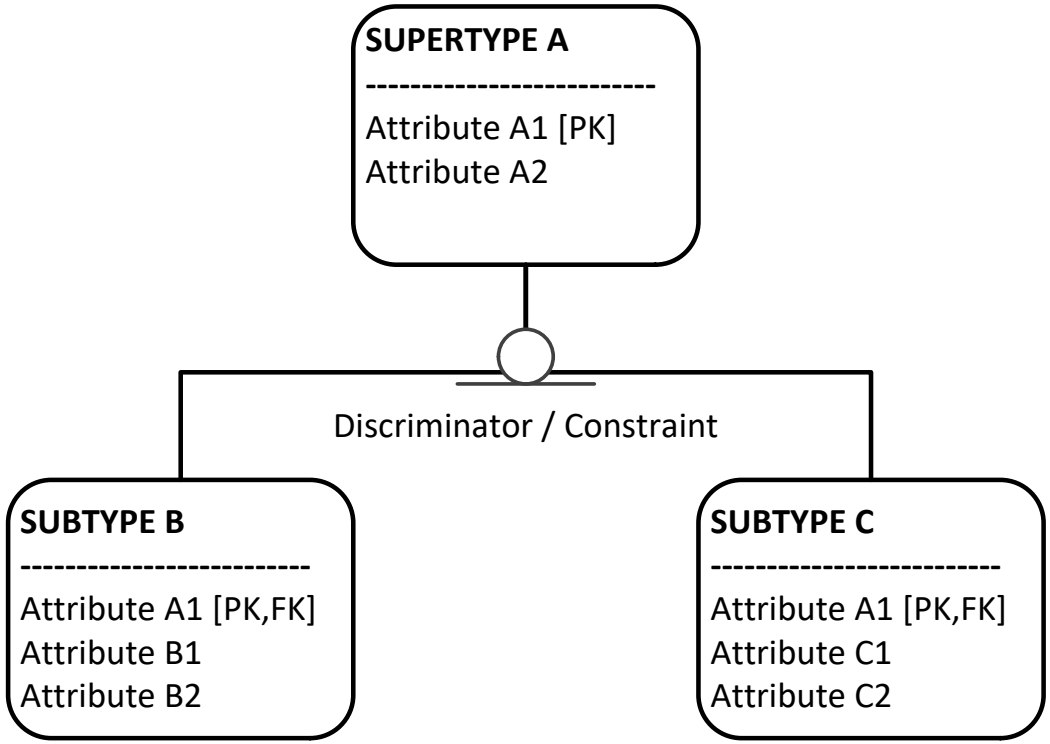


- **Total completeness**
  - Symbolized by a circle over a double line
  - Every supertype occurrence must be member of at least one subtype



Completeness Constraint

Type	Disjoint Constraint	Overlapping Constraint
Partial	• Supertype has optional subtypes. • Subtype discriminator can be null. • Subtype sets are unique.	• Supertype has optional subtypes. • Subtype discriminators can be null. • Subtype sets are not unique.
Total	• Every supertype occurrence is a member of only one subtype. • Subtype discriminator cannot be null. • Subtype sets are unique.	• Every supertype occurrence is a member of at least one subtype. • Subtype discriminators cannot be null. • Subtype sets are not unique.



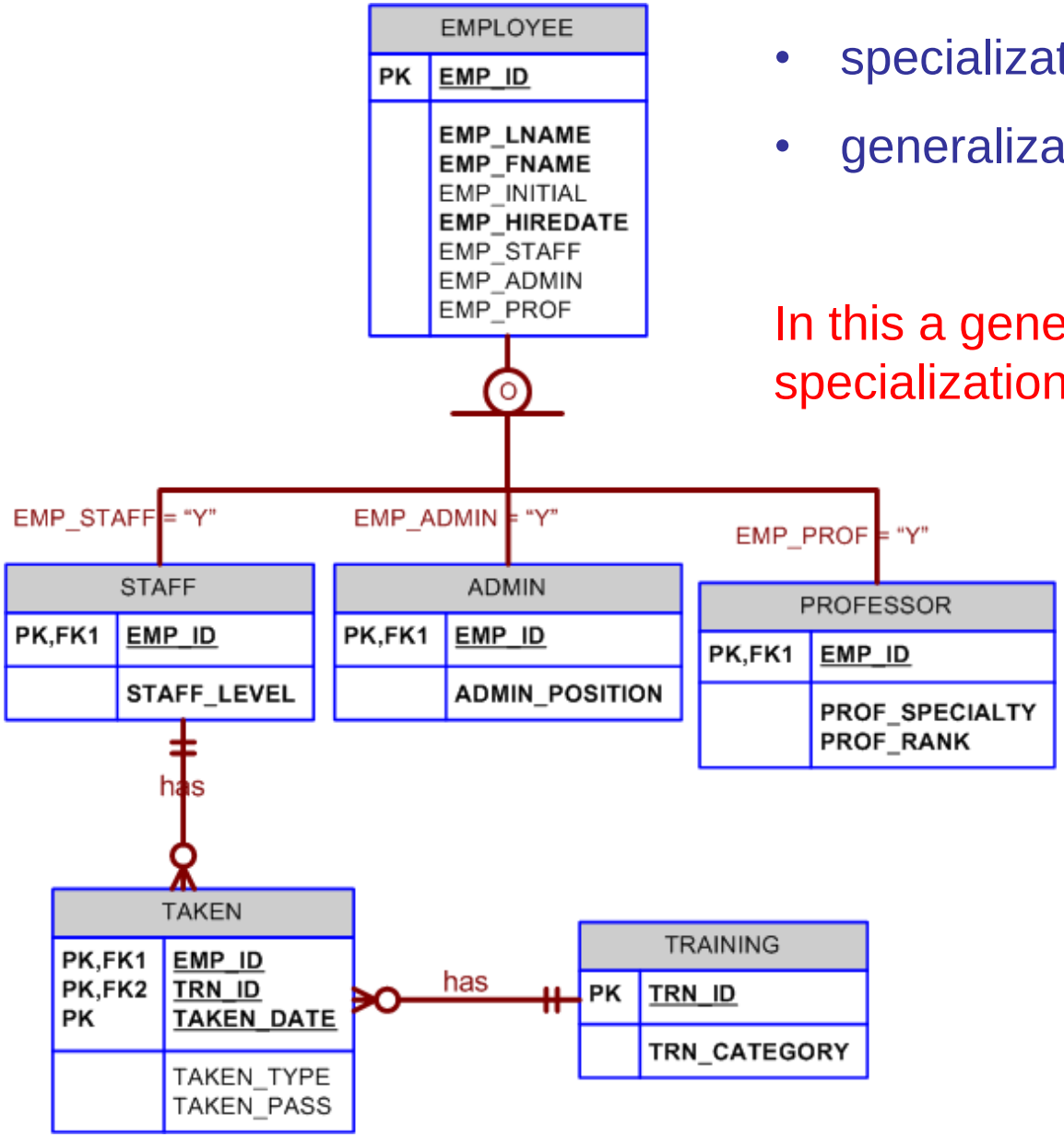
d	A1 + A2 + B1 + B2 **OR** A1 + A2 + C1 + C2
o	A1 + A2 + B1 + B2 + C1 + C2
d	A1 + A2 + B2 **OR** A1 + A2 + C1
o	A1 + A2 + B1 + C2

Specialization and Generalization

- **Specialization**
  - Identifies more specific entity subtypes from higher-level entity supertype
  - Top-down process
  - Based on grouping unique characteristics and relationships of the subtypes

Specialization and Generalization (cont'd.)

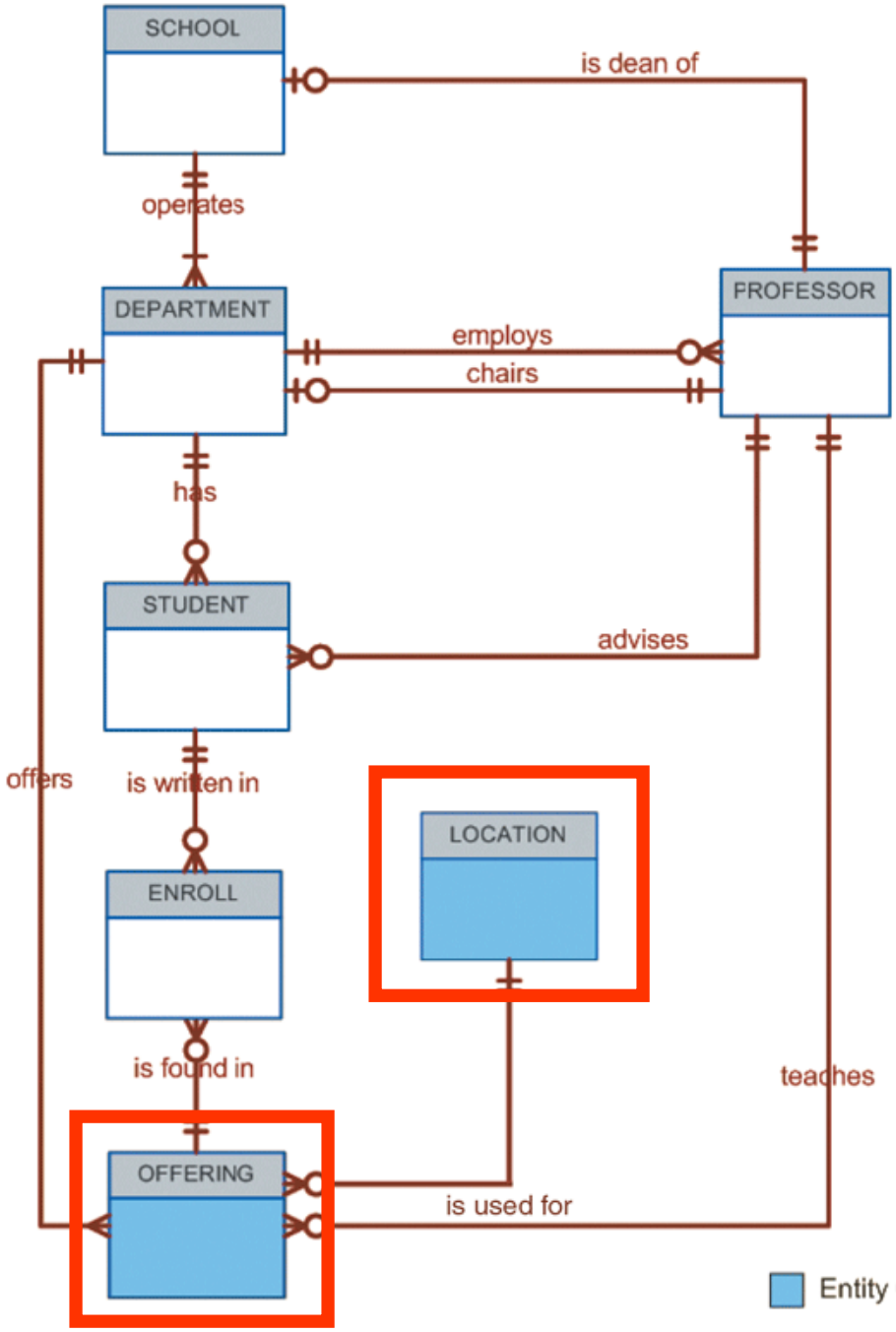
- **Generalization**
  - Identifies more generic entity supertype from lower-level entity subtypes
  - Bottom-up process
  - Based on grouping common characteristics and relationships of the subtypes



Most of the attributes if this was a

- specialization hierarchy?
- generalization hierarchy?

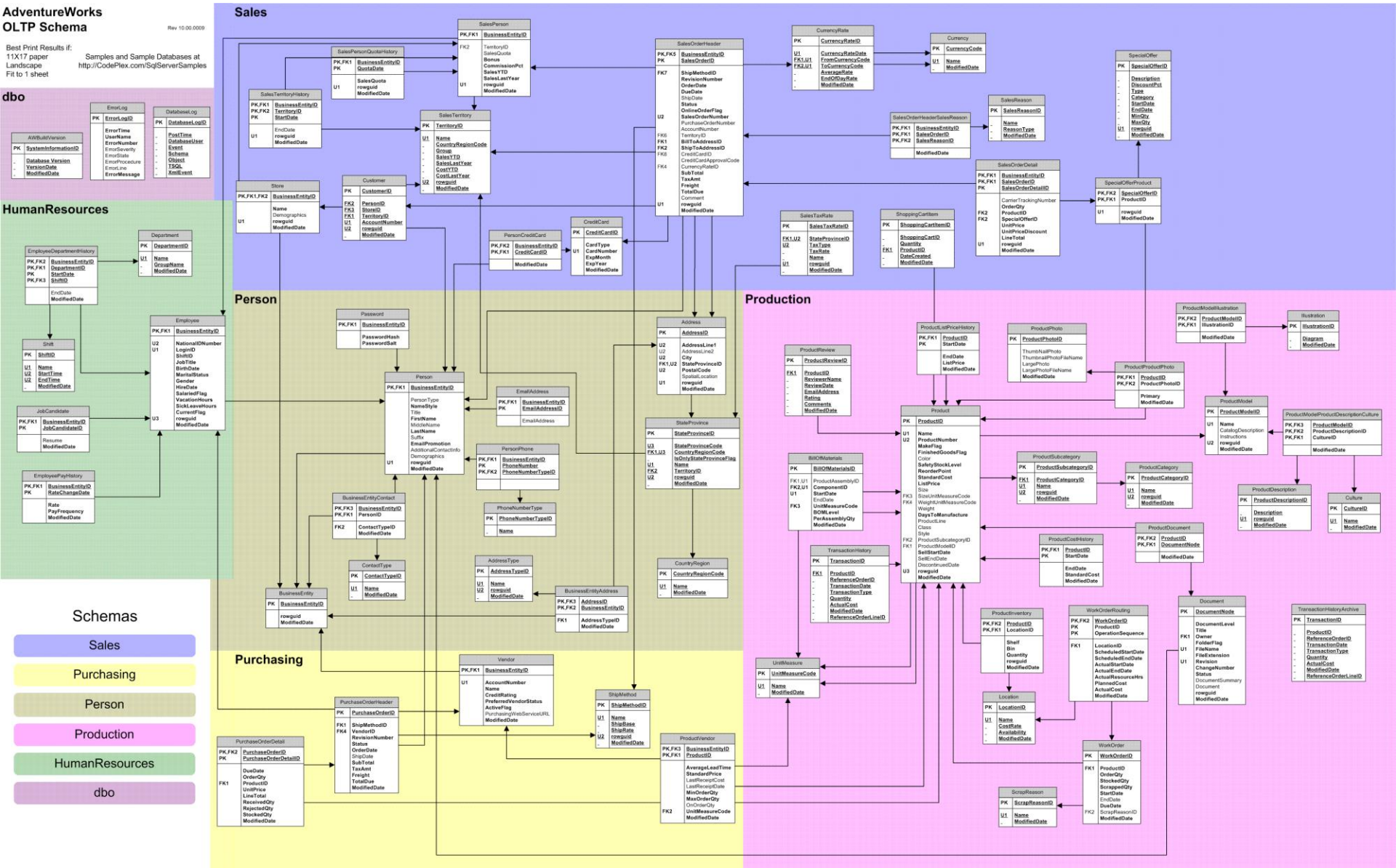
In this a generalization hierarchy or is it is specialization hierarchy?



Entity Clustering

- "Virtual" entity type used to represent multiple entities and relationships in ERD
- Considered "virtual" or "abstract" because it is not actually an entity in final ERD
- Temporary entity used to represent multiple entities and relationships
- Eliminate undesirable consequences
  - Avoid display of attributes when entity clusters are used

…… for example: The 6 clusters (schemas) of the AdventureWorks Sample Database



Entity Integrity: Selecting Primary Keys

- Primary key is the most important characteristic of an entity
  - Single attribute or some combination of attributes
- Primary key's function is to guarantee entity integrity
- Primary keys and foreign keys work together to implement relationships
- Properly selecting primary key has direct bearing on efficiency and effectiveness

Natural Keys and Primary Keys

- **Natural key** is a real-world identifier used to uniquely identify real-world objects
  - Familiar to end users and forms part of their day-to-day business vocabulary
- Generally, data modeler uses natural identifier as primary key of entity being modeled
- May instead use composite primary key or surrogate key

Primary Key Guidelines

- Attribute that uniquely identifies entity instances in an entity set
  - Could also be combination of attributes
- Main function is to uniquely identify an entity instance or row within a table
- Guarantee entity integrity, not to “describe” the entity
- Primary keys and foreign keys implement relationships among entities
  - Behind the scenes, hidden from user

When to Use Composite Primary Keys

- Composite primary keys useful in two cases:
  - As identifiers of composite entities
    - In which each primary key combination is allowed once in M:N relationship
  - As identifiers of weak entities
    - In which weak entity has a strong identifying relationship with the parent entity
- Automatically provides benefit of ensuring that there cannot be duplicate values

Example

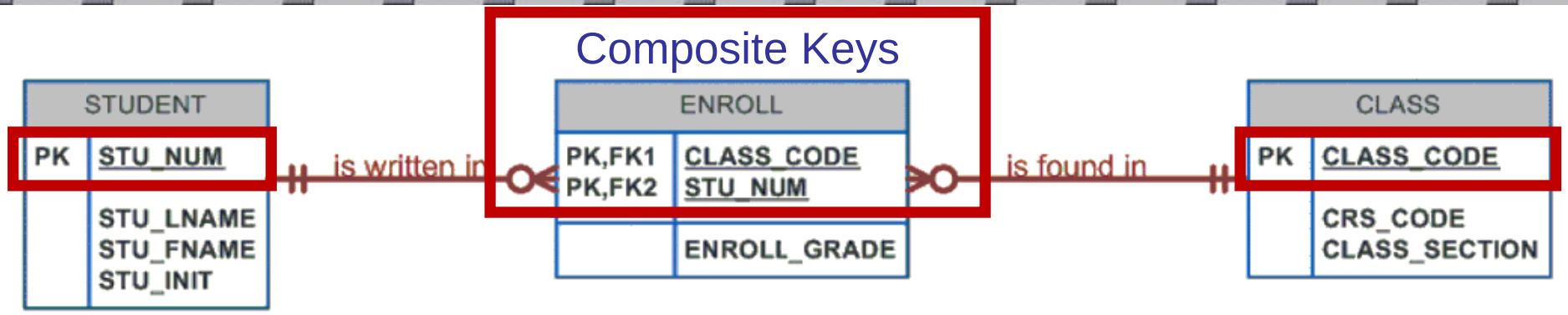


Table name: STUDENT
(first four fields)

STU_NUM	STU_LNAME	STU_FNAME	STU_INIT
321452	Bowser	William	C
324257	Smithson	Anne	K
324258	Brewer	Juliette	
324269	Oblonski	Walter	H
324273	Smith	John	D
324274	Katinga	Raphael	P
324291	Robertson	Gerald	T
324299	Smith	John	B

Table name: ENROLL

CLASS_CODE	STU_NUM	ENROLL_GRADE
10014	321452	C
10014	324257	B
10018	321452	A
10018	324257	B
10021	321452	C
10021	324257	C

Table name: CLASS
(first three fields)

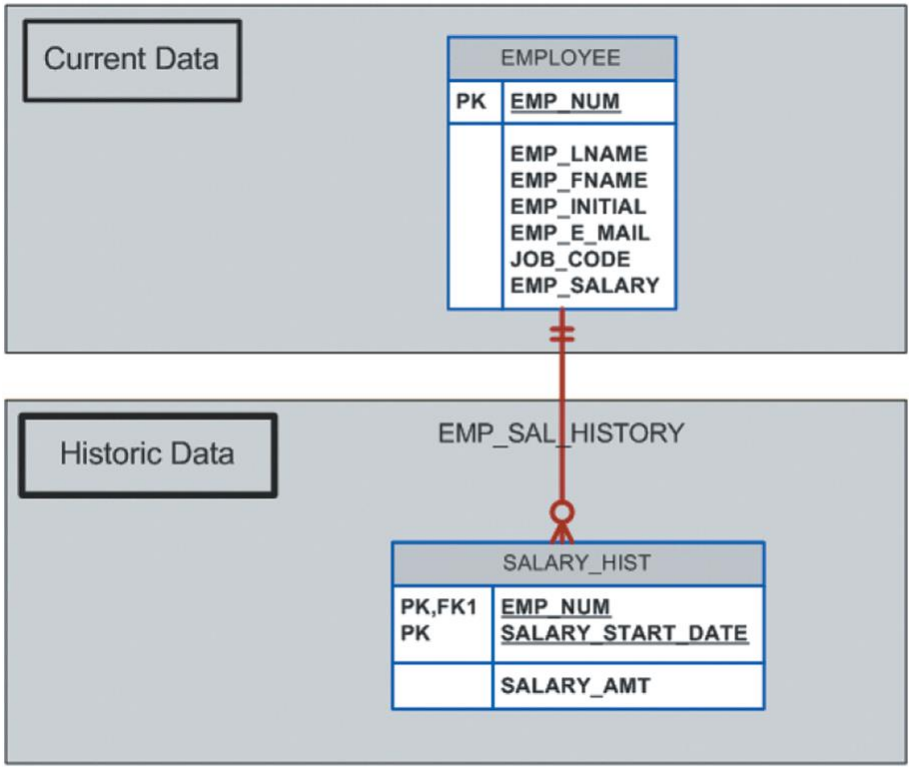
CLASS_CODE	CRS_CODE	CLASS_SECTION
10012	ACCT-211	1
10013	ACCT-211	2
10014	ACCT-211	3
10015	ACCT-212	1
10016	ACCT-212	2
10017	CIS-220	1
10018	CIS-220	2
10019	CIS-220	3
10020	CIS-420	1
10021	QM-261	1
10022	QM-261	2
10023	QM-362	1
10024	QM-362	2
10025	MATH-243	1

When to Use Composite Primary Keys (cont'd.)

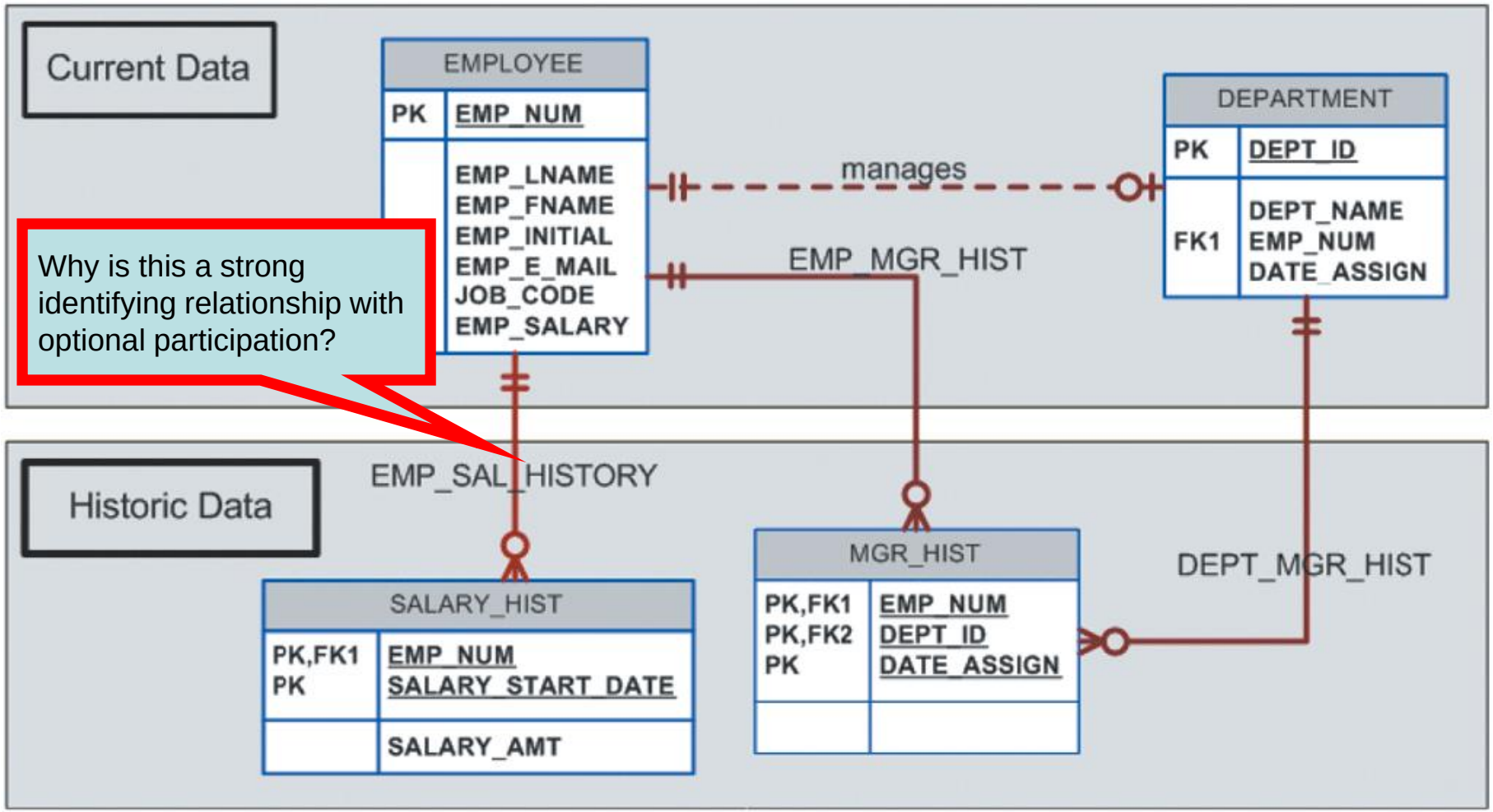
- When used as identifiers of weak entities normally used to represent:
  - Real-world object that is existent-dependent on another real-world object
  - Real-world object that is represented in data model as two separate entities in strong identifying relationship
- Dependent entity exists only when it is related to parent entity

Maintaining History of Time-Variant Data

- Time-variant data:
  - data whose values change over time and
  - a history of the data changes must be retained
  - Requires creating a new entity in a 1:M relationship with the original entity
  - New entity contains the new value, date of the change, and any other pertinent attribute



Example – History of Time Variant Data

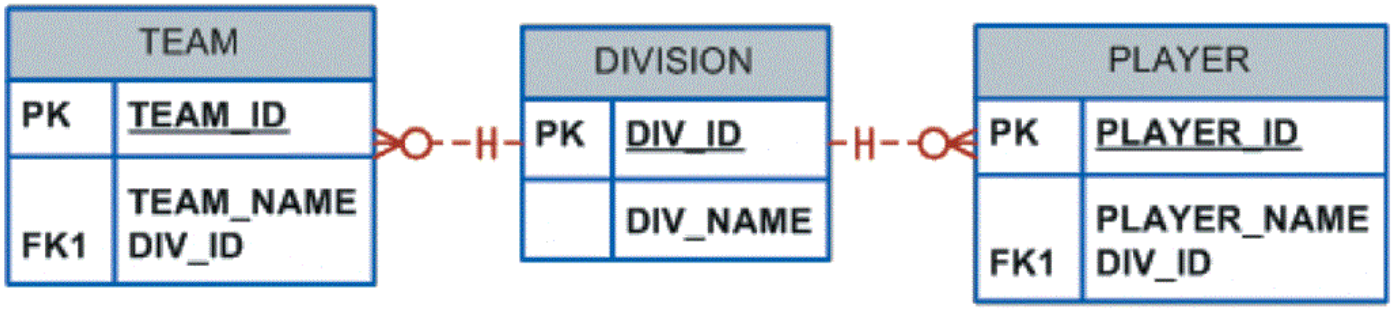


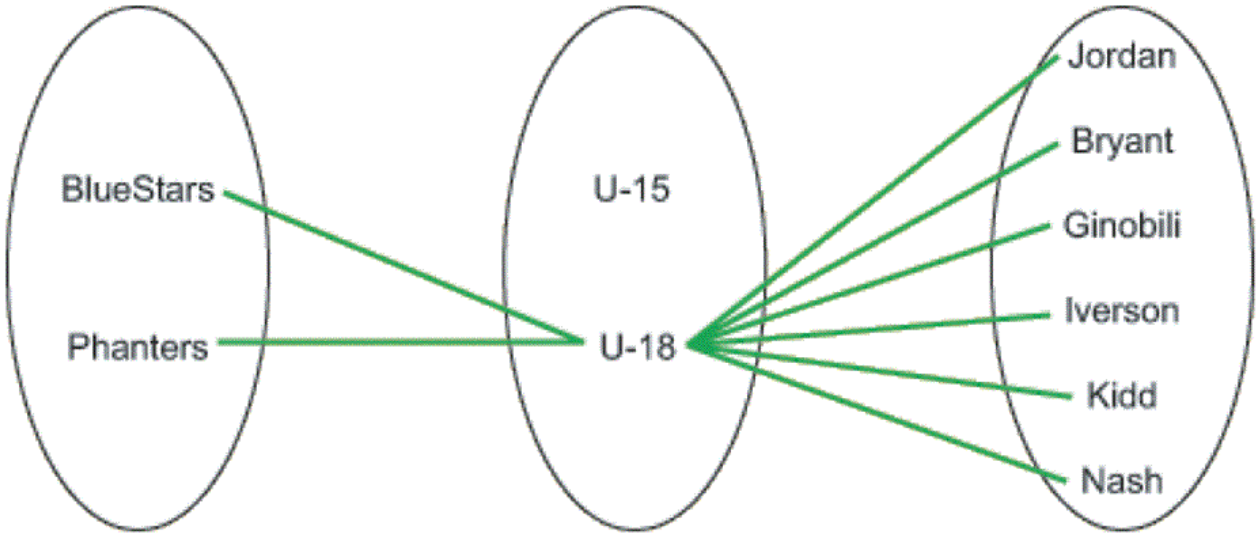
Something to look out for: Fan Traps and Design Traps

- **Design trap**: occurs when a relationship is improperly or incompletely identified
  - Represented in a way not consistent with the real world

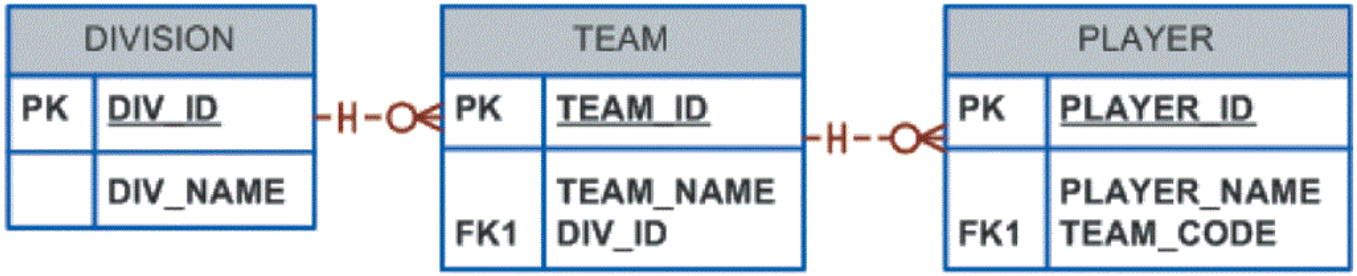
- **Fan trap**: occurs when one entity is in two 1:M relationships to other entities
  - Produces an association among other entities not expressed in the model

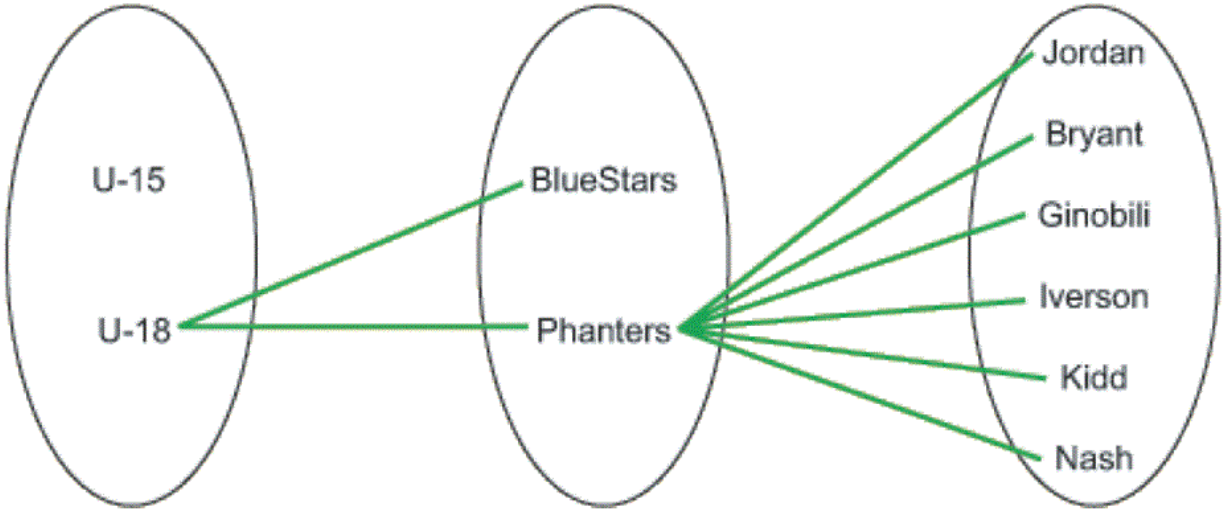
Fan Trap – Misidentified relationship





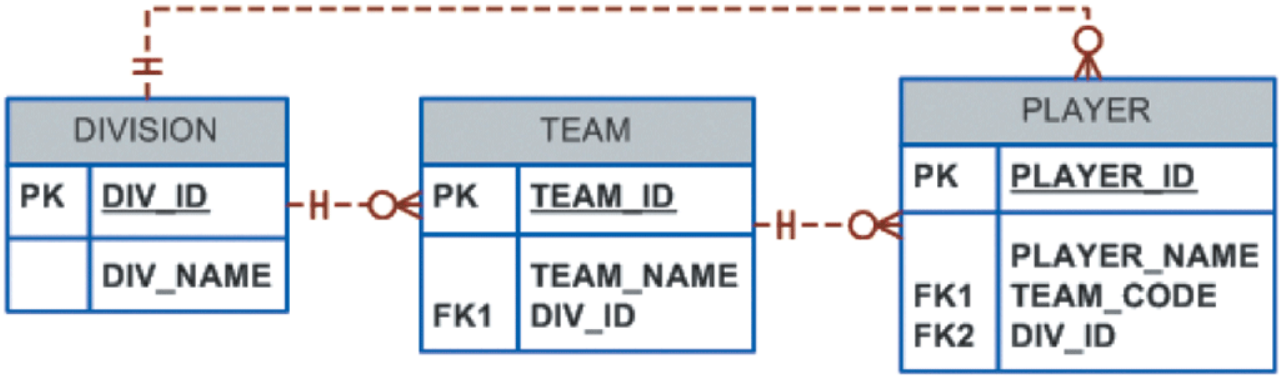
Fan trap eliminated





Something to look out for: Redundant Relationships

- Occur when there are multiple relationship paths between related entities
  - Must remain consistent across the model
  - Help simplify the design



QUESTIONS