Week 4

*Traps*

* Fan trap: Model suggests a relationship between two entities but relationship is ambiguous
* Chasm trap: Model suggests a relationship between two entities but pathway does not exist

*Logical Database Design*

* 7 steps:
  1. Get relations from EER
  2. Normalise relations
  3. Compare model with user transactions
  4. Integrity constraints
  5. Review logical data with user
  6. Merge logical model to global model
  7. Check future growth

*Relational Model Terminology*

* Relation: Logical model (not physical) is a table with rows and columns
* Attributes: Columns in the relation (fields)
* Domain: Set of values allowed for each attribute
* Tuples: Rows in the relation (uniquely-defined objects; records)
* Degrees: Number of attributes in a relation
* Cardinality: Number of tuples in a relation

*Integrity Constraints*

* Domain Constraints: Attributes must be atomic – no multi-valued or composite attributes
* Entity Integrity Constraints: No primary key can have a NULL value
* Referential Integrity Constraints: Each foreign key must match a primary key value in another relation, or NULL
* General Constraints: Other constraints related to business rules

*EER to Relational Mapping*

* 1:1 Relations
  + Mandatory on one side: If entity B only has mandatory participation with entity A on relation R, the foreign key is in B.
  + Mandatory on both sides: A & B are presented as a single relation
  + Optional on both sides: Foreign key can be in A or B.
* \*:\* Relations and n-ary Relationships
  + “Relationship” object with two (or more) foreign keys
* Multi-valued attributes and weak entities
  + Relation with foreign key
* Superclasses/subclasses
  + Mandatory “and”: One relation (with discriminants with discriminant for each tuple)
  + Optional “and”: Two relations; one for the superclass and one for the subclasses (with discriminants for each tuple)
  + Mandatory “or”: Many relations; one for each combined superclass and subclass
  + Optional “or”: Many relations; one for each superclass and subclass