1. **DB systems**
   1. Database: A structured and organized set of data accessible to people to which the data is relevant

DBMS: A software which has as its primary function to create a database on the specifics given as well as enabling maintenance of said database.

* + 1. Program-data independence: Changing the structure of the data does not necessarily require changing structure of all programs pertaining to the data set.
    2. Multiuser support: Supports concurrent multiple users. Ensures that concurrent updates of the same data is processed in an orderly manner, so as the data is always correct up to its latest update.
    3. Self-describing nature: The database system contains a definition of the database structure itself and its own constraints.

1. **ER model**
   * 1. An entity is a thing or object in the real world. An entity class is a set of entities which have similar attributes. As an example, consider Roger Midtstraum and Svein Erik Bratsberg. Independently they are each entities, which belong to the entity class “Teachers of TDT4145”.
     2. A relation is a reference between entities of differing entity class. A relation class defines a set of relations among the entity classes to which the relation class pertains to. In the already given example, Roger and Svein Erik has a relation WORKS\_FOR to NTNU, which is an entity of the “Employer” entity class. In this case the relation between the employees and employers all are contained in the relation class WORKS\_FOR.
     3. If an entity does not have a key attribute we have no consistent way of separating entities within the same entity class.

|  |  |  |
| --- | --- | --- |
| Statement # | Answer | Explanation |
| 1 | True | Taco is drawn up to be the entity class in the ER diagram, and the underlining of TacoID tells us it is the key attribute of taco. |
| 2 | True | The n-n relation between taco and its toppings tells ut this is true. |
| 3 | True | The lower bound of 0 on the relation of tacos in order confirms this. |
| 4 | True | The upper bound of n on the relation of tacos in order confirms this. |
| 5 | Maybe | This depends on the Order attribute “PickUpTime”, which may be instantaneously and it may not be. |
| 6 | False | The customer-Order relation demands an order, but not a customer, so if there is a customer there must be an order. |
| 7 | False | The entity does not have the weight, the relation does. |
| 8 | True | An employee must work at least one shop, but can work at more shops with different titles. |
| 9 | Maybe | There is no information on how delegation occurs. |
| 10 | False | A customer must not have a name registered. The only necessity is the key attribute CustomerNo |

1. **Weak entity types, functional data models and new demands**
   1. We use weak entity types when we have entities without key attributes on their own, and rely on their relation to other entities to produce a key attribute based on both some attribute of its own, and the relation it has to some other entity of a different entity type. In the case give, we have a weak entity class ScreenRoom which has a partial key, which means we cannot be certain it will give us only the correct entity. We also have a identifying relationship which together with the partial key gives a full identifier, for the entities of the ScreenRoom class.
   2. This will mean that the partial key and the relationship is not enough to secure identifier for the entity. The relationship is no longer an identifying one. We cannot have ScreenRoom as a weak entity type with the new cardinalities, as it will not provide a sufficient identification for each entity.

The rest of the questions are answered in diagrams which are in a separate PDF.