Data and Its Applications (CS4.301)

Monsoon 2021, IIIT Hyderabad
11 September, Monday (Lecture 7)

Taught by Prof. Kamal Karlapalem

Relational Data Model

Integrity Constraints

The data stored in the database must satisfy certain constraints at all times. These are domain constraints, key constraints, entity integrity constraints and referential integrity constraints.

- domain constraints – attributes A must always have values from dom(A), the set of atomic values permitted for A.
- key constraints – for any relation R over attributes $(A_1, A_2, \dots, A_n)$, the tuple $(A_{i_1}, A_{i_2}, \dots, A_{i_p})$ is a superkey sk if there are no two rows with the same value for these attributes. A key is a minimal is a minimal superkey, *i.e.*, no attribute can be removed from it while still keeping it as a superkey. Every relationship must have a key.
- entity integrity constraints – if S is a relational database, its schema is $S = \{R_1, R_2, \dots, R_n\}$. The primary keys pk of each $R_i \in S$ cannot have null values in any tuple.
- referential integrity constraints – the pk of a certain relation (the referenced relation) can be an attribute in another relation (the referencing relation), in which case it is a foreign key or fk in the latter. Then the values taken by the fk must either exist in the relation for which it is a pk or be null.

At every update (insertion, deletion or modification) of the database, the system should check that none of the constraints are violated. In case of a violation, the system could

- cancel the operation
- perform it but inform the user
- trigger additional updates to correct it
- execute a user-specified error-correction routine

ER-Relational Correspondence

Given the schema of an ER database, it can be implemented in a relational data model by

- creating a relation for each strong entity type
- for each binary relationship:
  - if it is 1:1, adding it as an attribute to either of the tables for the entity types.
  - if it is 1:N, adding it as an attribute to the table for the entity type on the N side.
  - if it is M:N, creating a new table with the primary keys of both entity types.
- creating a relation for each weak entity type, using the partial key and the primary key of the owner entity type.
- creating a relation for each higher-degree relationship using the primary keys of all participating entity types.
- creating a relation for all specialisations (subclasses) giving the extra attributes with the primary key of the superclass. If the entity has total participation, the relation for the superclass as a whole is not needed; the attributes can be added to the subclass relations. Alternatively, there can be an attribute in the main tabe indicating which type the entity falls in (or a flag for each type), and columns for the attributes of each type.