Mapping of ER Diagrams to Database Schema

With the ER diagrams constructed, we can now follow a series of steps to convert the ER diagram to database schema.

|  |
| --- |
| **Step 1**  Mapping of Regular Entity Types  For each of the regular entity type (Not weak entities), create a relation (table) for each one of them, that includes **all of the simple attributes** of that entity.  This means, we are not going to include any foreign keys or relationship attributes, ONLY SIMPLE ATTRIBUTES |
| **Step 2**  Mapping of Weak Entity Types  For each weak entity type, create a relation (table) that **includes all the simple attributes** of the weak entity  In addition to that, since weak entities don’t exist if the owner entity doesn’t, **we set up the primary keys to be composite** (Composite primary key, where 2 or more fields are primary keys, forming a combination)  **The primary key shall be the partial key of the weak entity, plus the primary key of its owner entity**  Eg: Both the exam\_id and the class\_id shall both form a composite primary key |
| **Step 3**  Mapping of Binary 1 – 1 Relationship Types  For those entities connected together by 1 – 1 Relationship cardinality, **Pick one of them to include the primary key of the other as foreign key.**  When picking, we **FAVOR the TOTAL PARTICIPATION SIDE to have the foreign key field instead of the one with Partial Participation**  Eg: Imagine that  **| Branch |====< Manages >------| Employee |**  The relationship here is that Branch must have one employee as its manager.  If we put the foreign key field in Employee, The employee may or may not be manager. This will result in a lot of NULL data!  Instead, since Branch MUST have one employee as its manager, it makes sense to put the foreign key field there, since there shall be no NULL data inside. |
| **Step 4**  Mapping of Binary 1 – N Relationship Types  We need to include the 1 side’s primary key as the foreign key on the N side  (Meaning, N side entity will get added a foreign key field)  Eg:  **N 1**  **| Employee |=======< Works At >========| Branch |**  Each branch may have a lot of employees! It’s unrealistic to add a ton of foreign keys of employee in one entry of branch!  Therefore, since each employee works at only one branch, each employee having a branch\_id he is working at is much more suitable |
| **Step 5**  Mapping of Binary N – M Relationship Types  In that case, we shall create a new table (relation) which has composite primary keys, where the primary key consists of foreign keys from both entities  Eg:  **N M**  **| Employee |======< Working >=======| Client |**  Each employee can work with multiple clients, and each client can work with multiple employees. Therefore, we should put those foreign key pairs in a new table, with those combination representing a primary key itself |