RELATIONAL MODEL

1. Relation Schema and Relation Instance

- **A relation schema** is a list of attribute names (user defined domain names), e.g. Account = (branch-name, account#, balance). However, for various reasons, a relation schema is often represented by a set (not an ordered list) of attributes.

- **A relation instance** is a subset of a Cartesian product of a list of domains.
  
  e.g., account(Account): "account" is a relation instance on "Account" schema. The relation instance "account" is a subset of D1 × D2 × D3, where D1, D2, and D3 are the domains of "branch-name", "account-name", and "balance", respectively.

- **A tuple** is an element of a relation instance. If there are n attributes, each tuple of the relation instance is an ordered n-tuple. Thus, each tuple is an n-ary relationship between n attribute values. A relation is, in fact, a set of such relationships.

- "relation" often means "relation instance"
• "database" often means "database instance"

• Recall the three-level database (Lecture Note#1). In the logical level, database schema is a set of relation schemas and database instance is a set of relation instances.

Note, in relational data model, relation schema consists of atomic domains. A domain is atomic if each element of the domain is conceptually indivisible. The balance of an account is atomic. However, the domains of "date" or "author-list" of a book are not atomic.

c.f. Object-Relational DB allows complex data types (i.e., non-atomic domains).

• **Keys:** Superkey, Candidate key, Primary key, and Foreign key.

  *Candidate key:* a set of attributes whose values uniquely identify individual tuples in the relation (e.g., \{Social Security\#}, \{Employee\#\}, \{Employee\#, Dependent Name\}). None of its subset can be used to uniquely identify tuples (e.g., \{Employee\#\} or \{Dependent Name\} cannot uniquely identify a tuple in the "Dependent" table). "Null" value is not allowed. Each tuple has a unique candidate key value in the relation.
**Primary key**: one of the candidate keys of the relation, chosen by the designer. "Null" value is not allowed (i.e., all attributes of the primary key are implicitly declared to be not null). Each tuple has a unique primary key value in the relation. If an attribute set P is the primary key of relation r, then P must be a candidate key of r. The opposite does not hold.

**Super key**: union of (one or more candidate keys) and (zero, one, or more extraneous attributes). If an attribute set C is a candidate key of relation r, then C must be a super key of r. The opposite does not hold.

**Foreign key**: We use this key to represent the relationship between relations. Let r(R) and s(S) be relations with primary keys $K_r \subseteq R$ and $K_s \subseteq S$, respectively. If $F_r \subseteq R$ is a foreign key referencing $K_s$, for every $t_1 \in r$ with $t_1[F_r] \neq \text{null}$, there exist a tuple $t_2 \in s$ such that $t_1[F_r] = t_2[K_s]$. Note, the mapping cardinality between r and s must be one-to-one or many-to-one (note, r is on the many side). Note, $t_1[F_r]$ is the foreign key value of tuple $t_1$ and $t_2[K_s]$ is the primary key value of tuple $t_2$.

- **A Query Language** is a language in which users request information from the underlying database. (Relational Algebra, Relational Calculus, QBE, SQL, …)