The Relational Data Model and Relational Database Constraints

Part 2

Relational Model Constraints and Relational Database Schemas

Part 1

- Constraints on databases can generally be divided into three main categories:
  - Constraints that are inherent in the data model. We call these **inherent model-based constraints** or **implicit constraints**.
  - Constraints that can be directly expressed in schemas of the data model, typically by specifying them in the DDL (data definition language). We call these **schema-based constraints** or **explicit constraints**.
  - Constraints that *cannot* be directly expressed in the schemas of the data model, and hence must be expressed and enforced by the application programs. We call these **application-based** or **semantic constraints** or business rules.

- The characteristics of relations that we discussed earlier are the inherent constraints of the relational model and belong to the first category.
- For example, the constraint that a relation cannot have duplicate tuples is an inherent constraint.

- The schema-based constraints include domain constraints, key constraints, constraints on NULLs, entity integrity constraints, and referential integrity constraints.

Domain Constraints

- Domain constraints specify that within each tuple, the value of each attribute *A* must be an atomic value from the domain dom(*A*).

Key Constraints and Constraints on NULL Values

- A **superkey** SK specifies a *uniqueness constraint* that no two distinct tuples in any state *r* of *R* can have the same value for SK.
- Every relation has at least one default superkey—the set of all its attributes.

- A superkey can have redundant attributes, however, so a more useful concept is that of a *key*, which has no redundancy.

- A **key** *K* of a relation schema *R* is a superkey of *R* with the additional property that removing any attribute *A* from *K* leaves a set of attributes *K'* that is not a superkey of *R* any more.
- Hence, a key satisfies two properties:
  1. Two distinct tuples in any state of the relation cannot have identical values for (all) the attributes in the key. This first property also applies to a superkey.
  2. It is a *minimal superkey*—that is, a superkey from which we cannot remove any attributes and still have the uniqueness constraint in condition 1 hold. This property is not required by a superkey.

- Notice that a set of attributes constituting a key is a property of the relation schema; it is a constraint that should hold on *every* valid relation state of the schema.
- A key is determined from the meaning of the attributes, and the property is *time-invariant*: It must continue to hold when we insert new tuples in the relation.

- In general, a relation schema may have more than one key.
- In this case, each of the keys is called a **candidate key**.
- It is common to designate one of the candidate keys as the **primary key** of the relation.
- We use the convention that the attributes that form the primary key of a relation schema are underlined.

CAR

License_number	Engine_serial_number	Make	Model	Year
Texas ABC-739	A69352	Ford	Mustang	02
Florida TVP-347	B43696	Oldsmobile	Cutlass	05
New York MPO-22	X83554	Oldsmobile	Delta	01
California 432-TFY	C43742	Mercedes	190-D	99
California RSK-629	Y82935	Toyota	Camry	04
Texas RSK-629	U028365	Jaguar	XJS	04

- Notice that when a relation schema has several candidate keys, the choice of one to become the primary key is somewhat arbitrary; however, it is usually better to choose a primary key with a single attribute or a small number of attributes.
- The other candidate keys are designated as **unique keys**, and are not underlined.

- Another constraint on attributes specifies whether NULL values are or are not permitted.