**Entity Integrity constraints, Referential Integrity constraints, and Foreign Keys**

**Entity Integrity constraints:**

The entity integrity constraint states that no primary key value can be NULL. This is because the primary key value is used to identify individual tuples in a relation. Having NULL values for the primary key implies that we cannot identify some tuples.

For example, if two or more tuples had NULL for their primary keys, we may not be able to distinguish them if we try to reference them from other relations.

**Referential Integrity constraints:**

The referential integrity constraint is specified between two relations and is used to maintain the consistency among tuples in the two relations. Informally, the referential integrity constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation.

For example, in Figure 5.6, the attribute Dno of EMPLOYEE gives the department number for which each employee works; hence, its value in every EMPLOYEE tuple must match the Dnumber value of some tuple in the DEPARTMENT relation.

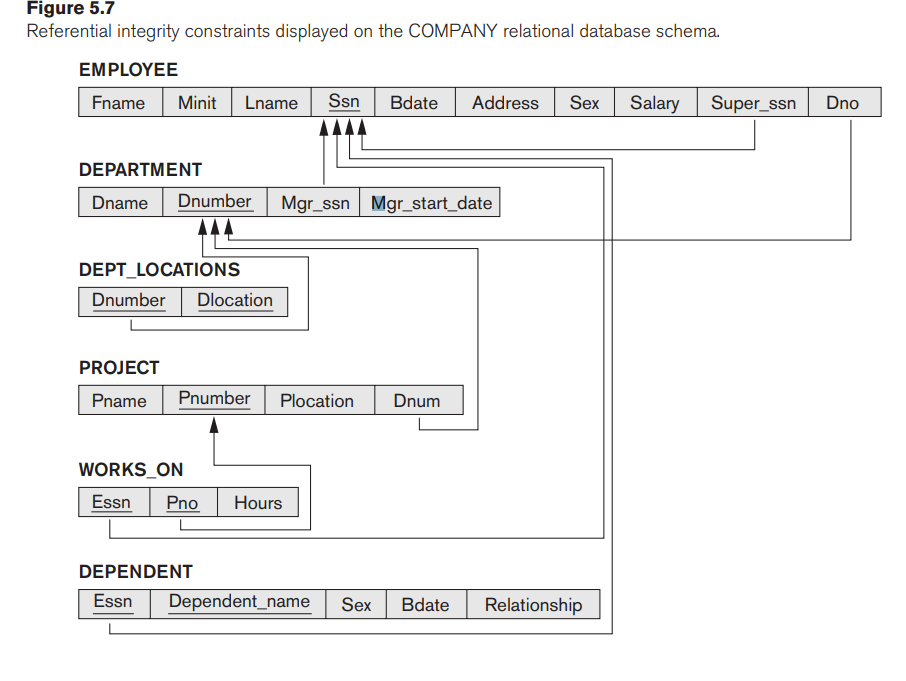
To define referential integrity more formally, first we define the concept of a foreign key.

The conditions for a foreign key, given below, specify a referential integrity constraint between the two relation schemas R1 and R2. A set of attributes FK in relation schema R1 is a foreign key of R1 that references relation R2 if it satisfies the following rules:

1. The attributes in FK have the same domain(s) as the primary key attributes PK of R2; the attributes FK are said to reference or refer to the relation R2.

2. A value of FK in a tuple t1 of the current state r1(R1) either occurs as a value of PK for some tuple t2 in the current state r2(R2) or is NULL. In the former case, we have t1[FK] = t2[PK], and we say that the tuple t1 references or refers to the tuple t2.

In this definition, R1 is called the referencing relation and R2 is the referenced relation. If these two conditions hold, a referential integrity constraint from R1 to R2 is said to hold.



In SQL, the CREATE TABLE statement of the SQL DDL allows the definition of primary key,

unique key, NOT NULL, entity integrity, and referential integrity constraints.

There are three basic operations that can change the states of relations in the database: Insert, Delete, and Update (or Modify). They insert new data, delete old data, or modify existing data records, respectively.

Insert is used to insert one or more new tuples in a relation, Delete is used to delete tuples, and Update (or Modify) is used to change the values of some attributes in existing tuples.

Whenever these operations are applied, the integrity constraints specified on the relational database schema should not be violated.

**The Insert Operation**

The Insert operation provides a list of attribute values for a new tuple t that is to be inserted into a relation R.

Operation:

Insert <‘Cecilia’, ‘F’, ‘Kolonsky’, NULL, ‘1960-04-05’, ‘6357 Windy Lane, Katy,TX’, F, 28000, NULL, 4> into EMPLOYEE.

Result: This insertion violates the entity integrity constraint (NULL for the primary key Ssn), so it is rejected.

■ Operation:

Insert <‘Alicia’, ‘J’, ‘Zelaya’, ‘999887777’, ‘1960-04-05’, ‘6357 Windy Lane, Katy, TX’, F, 28000, ‘987654321’, 4> into EMPLOYEE.

Result: This insertion violates the key constraint because another tuple with the same Ssn value already exists in the EMPLOYEE relation, and so it is rejected.

■ Operation:

Insert <‘Cecilia’, ‘F’, ‘Kolonsky’, ‘677678989’, ‘1960-04-05’, ‘6357 Windswept, Katy, TX’, F, 28000, ‘987654321’, 7> into EMPLOYEE.

Result: This insertion violates the referential integrity constraint specified on Dno in EMPLOYEE because no corresponding referenced tuple exists in DEPARTMENT with Dnumber = 7.

Operation:

Insert <‘Cecilia’, ‘F’, ‘Kolonsky’, ‘677678989’, ‘1960-04-05’, ‘6357 Windy Lane, Katy, TX’, F, 28000, NULL, 4> into EMPLOYEE.

Result: This insertion satisfies all constraints, so it is acceptable

**The Delete Operation**

The Delete operation can violate only referential integrity. This occurs if the tuple being deleted is referenced by foreign keys from other tuples in the database. To specify deletion, a condition on the attributes of the relation selects the tuple (or tuples) to be deleted. Here are some examples.

■ Operation:

Delete the WORKS\_ON tuple with Essn = ‘999887777’ and Pno = 10.

Result: This deletion is acceptable and deletes exactly one tuple.

■ Operation:

Delete the EMPLOYEE tuple with Ssn = ‘999887777’.

Result: This deletion is not acceptable, because there are tuples in WORKS\_ON that refer to this tuple. Hence, if the tuple in EMPLOYEE is deleted, referential integrity violations will result.

■ Operation:

Delete the EMPLOYEE tuple with Ssn = ‘333445555’.

Result: This deletion will result in even worse referential integrity violations, because the tuple involved is referenced by tuples from the EMPLOYEE, DEPARTMENT, WORKS\_ON, and DEPENDENT relations.

**The Update Operation**

The Update (or Modify) operation is used to change the values of one or more attributes in a tuple (or tuples) of some relation R. It is necessary to specify a condition on the attributes of the relation to select the tuple (or tuples) to be modified.

Here are some examples.

■ Operation:

Update the salary of the EMPLOYEE tuple with Ssn = ‘999887777’ to 28000.

Result: Acceptable.

■ Operation:

Update the Dno of the EMPLOYEE tuple with Ssn = ‘999887777’ to 1.

Result: Acceptable.

■ Operation:

Update the Dno of the EMPLOYEE tuple with Ssn = ‘999887777’ to 7.

Result: Unacceptable, because it violates referential integrity.

■ Operation:

Update the Ssn of the EMPLOYEE tuple with Ssn = ‘999887777’ to ‘987654321’.

Result: Unacceptable, because it violates primary key constraint by repeating a value that already exists as a primary key in another tuple; it violates referential integrity constraints because there are other relations that refer to the existing value of Ssn.