Normalization

- Normalization is essentially a two step process that puts data into tabular form by removing repeating groups and then removes duplicated data from the relational tables.
- Normalization theory is based on the concepts of **normal forms**. A relational table is said to be a particular normal form if it satisfied a certain set of constraints

Benefits Of Normalization

- Data Consistency
- Reduction In Storage Space
- Organized data

Functional Dependencies

- A column, Y, of the relational table R is said to be **functionally dependent** upon column X of R if and only if each value of X in R is associated with precisely one value of Y at any given time. X and Y may be composite.
- Saying that column Y is functionally dependent upon X is the same as saying the values of column X identify the values of column Y. If column X is a primary key, then all columns in the relational table R must be functionally dependent upon X.
- A short-hand notation for describing a functional dependency is:

  R.x ->; R.y

  which can be read as in the relational table named R, column x functionally determines (identifies) column y.

Full Functional Dependence

- Full functional dependence applies to tables with composite keys. Column Y in relational table R is fully functional on X of R if it is functionally dependent on X and not functionally dependent upon any subset of X.

- Full functional dependence means that when a primary key is composite, made of two or more columns, then the other columns must be identified by the entire key and not just some of the columns that make up the key.

First Normal Form

- Requirements For 1NF:
  - The data should be in *tabular format.*
  - For each row, each column should have a *single value*.
  - There should be *no duplicate rows.*

First Normal Form

FIRST

s#	status	city	p#	qty
s1	20	London	p1	300
s1	20	London	p2	200
s1	20	London	p3	400
s1	20	London	p4	200
s1	20	London	p5	100
s1	20	London	p6	100
s2	10	Paris	p1	300
s2	10	Paris	p2	400
s3	10	Paris	p2	200
s4	20	London	p2	200
s4	20	London	p4	300
s4	20	London	p5	400

Although the table FIRST is in 1NF it contains redundant data. For example, information about the supplier's location and the location's status have to be repeated for every part supplied.Redundancy causes what are called *update anomalies*.Update anomalies are problems that arise when information is inserted, deleted, or updated.

First Normal Form(cont…)

- For example, the following anomalies could occur in FIRST:
- INSERT. The fact that a certain supplier (s5) is located in a particular city (Athens) cannot be added until they supplied a part.
- DELETE. If a row is deleted, then not only is the information about quantity and part lost but also information about the supplier.
- UPDATE. If supplier s1 moved from London to New York, then six rows would have to be updated with this new information.

Second Normal Form

- **Def:** A relational table is in second normal form 2NF if it is in 1NF and every non-key column is fully dependent upon the main key that uniquely identifies the column(*primary key*).

Second Normal Form(cont…)

- FIRST is in 1NF but not in 2NF because status and city are functionally dependent upon only on the column s# of the composite key (s#, p#).

FIRST

s#	status	city	p#	qty
s1	20	London	p1	300
s1	20	London	p2	200
s1	20	London	p3	400
s1	20	London	p4	200
s1	20	London	p5	100
s1	20	London	p6	100
s2	10	Paris	p1	300
s2	10	Paris	p2	400
s3	10	Paris	p2	200
s4	20	London	p2	200
s4	20	London	p4	300
s4	20	London	p5	400

Second Normal Form(cont…)

The process for transforming a 1NF table to 2NF is:

- Identify any determinants (which determine some columns) other than the main key and the columns they determine.
- Create and name a new table for each determinant and the columns it determines.
- Move the determined columns from the original table to the new table. The determinant becomes the primary key of the new table.
- Delete the columns you just moved from the original table except for the determinant which will serve as a foreign key.
- The original table may be renamed to maintain semantic meaning.

Second Normal Form(cont…)

- To transform FIRST into 2NF we move the columns s#, status, and city to a new table called SECOND. The column s# becomes the primary key of this new table.
- Tables in 2NF but not in 3NF still contain modification anomalies. In the example of SECOND, they are:
- INSERT. The fact that a particular city has a certain status (Rome has a status of 50) cannot be inserted until there is a supplier in the city.
- DELETE. Deleting any row in SUPPLIER destroys the status information about the city as well as the association between supplier and city.

Second Normal Form(cont…)

SECOND

s#	status	city
s1	20	London
s2	10	Paris
s3	10	Paris
s4	20	London
s5	30	Athens

PARTS

s#	p#	qty
s1	p1	300
s1	p2	200
s1	p3	400
s1	p4	200
s1	p5	100
s1	p6	100
s2	p1	300
s2	p2	400
s3	p2	200
s4	p2	200
s4	p4	300
s4	p5	400

Third Normal Form (3NF)

- A relational table is in third normal form (3NF) if it is already in 2NF and every non-key column is non transitively dependent upon its primary key.
- The third normal form requires that all columns in a relational table are dependent only upon the primary key.

Third Normal Form (3NF) Contd...

- A *transitive dependency* is occurs when a non-key column that is a determinant of the primary key is the determinate of other columns.
- Table PARTS is already in 3NF. The non-key column, qty, is fully dependent upon the primary key (s#, p#). SUPPLIER is in 2NF but not in 3NF because it contains a *transitive dependency*.

PARTS

s#	p#	qty
s1	p1	300
s1	p2	200
s1	p3	400
s1	p4	200
s1	p5	100
s1	p6	100
s2	p1	300
s2	p2	400
s3	p2	200
s4	p2	200
s4	p4	300
s4	p5	400

SECOND

s#	status	city
s1	20	London
s2	10	Paris
s3	10	Paris
s4	20	London
s5	30	Athens

Third Normal Form (3NF) Contd...

- The concept of a transitive dependency can be illustrated by showing the functional dependencies in SUPPLIER:

  SUPPLIER.s# -> SUPPLIER.status

  SUPPLIER.s# -> SUPPLIER.city

  SUPPLIER.city -> SUPPLIER.status

- Note that SUPPLIER.status is determined both by the primary key s# and the non-key column city.

Third Normal Form (3NF) Contd...

- The process of transforming a table into 3NF is:
    - Identify any determinants, other the primary key, and the columns they determine.
    - Create and name a new table for each determinant and the unique columns it determines.
    - Move the determined columns from the original table to the new table. The determinate becomes the primary key of the new table.
    - Delete the columns you just moved from the original table except for the determinate which will serve as a foreign key.
    - The original table may be renamed to maintain semantic meaning.

Third Normal Form (3NF) Contd...

- To transform SUPPLIER into 3NF, we create a new table called CITY_STATUS and move the columns city and status into it.
- Status is deleted from the original table, city is left behind to serve as a foreign key to CITY_STATUS, and the original table is renamed to SUPPLIER_CITY to reflect its semantic meaning.

Third Normal Form (3NF) Contd...

- The results are shown in Figure 3 below.
  - Figure 3: Tables in 3NF

SUPPLIER_CITY

s#	city
s1	London
s2	Paris
s3	Paris
s4	London
s5	Athens

CITY_STATUS

city	status
London	20
Paris	10
Athens	30
Rome	50

Third Normal Form (3NF) Contd...

- The results of putting the original table into 3NF has created three tables. These can be represented in "psuedo-SQL" as:
  - PARTS (#s, p#, qty)
    Primary Key (s#,#p)
    Foreign Key (s#) references SUPPLIER_CITY.s#
  - SUPPLIER_CITY(s#, city)
    Primary Key (s#)
    Foreign Key (city) references CITY_STATUS.city
  - CITY_STATUS (city, status)
    Primary Key (city)

Boyce Codd Normal Form

- BCNF - A relation is in BCNF if and only if every determinant is a candidate key(*column or group of columns that uniquely identifies a row*).
- Boyce-Codd normal form (BCNF) is a more rigorous version of the 3NF which deals with relational tables that had.
  - (a) multiple candidate keys,
  - (b) composite candidate keys, and
  - (c) candidate keys that overlapped.
- Every relation in BCNF is also in 3NF. However, relation in 3NF may not be in BCNF.

Boyce Codd Normal Form

- Difference between 3NF and BCNF is that for a functional dependency A $\rightarrow$ B, 3NF allows this dependency in a relation if B is a primary-key attribute and A is not a candidate key.
- Whereas, BCNF insists that for this dependency to remain in a relation, A must be a candidate key.

Boyce Codd Normal Form

- Violation of BCNF is quite rare.

- Potential to violate BCNF may occur in a relation that:
  - contains two (or more) composite candidate keys;
  - the candidate keys overlap (i.e. have at least one attribute in common).

Explaining BCNF

- Consider the previous example, suppose we consider that supplier names are unique and also columns city and status are mutually independent.
- Now, we consider following relation

  SSP { S#,SNAME,P#,QTY }

  The candidate keys are { S#,P# } and {SNAME,P# } .

  This relation is not in BCNF.
- We convert the relation into BCNF by decomposing the relation in either of the ways given below :

  SS { S#,SNAME } and SP { S#, P#, QTY } OR

  SS { S#,SNAME } and SP {SNAME, P#, QTY } .

Fourth Normal Form (4NF)

- Although BCNF removes anomalies due to functional dependencies, another type of dependency called a *multi-valued dependency (MVD)* can also cause data redundancy.

- Possible existence of MVDs in a relation is due to 1NF and can result in data redundancy.

Fourth Normal Form (4NF) : *Multi-Valued Dependency.*

- Dependency between attributes (for example, A, B, and C) in a relation, such that for each value of A there is a set of values for B and a set of values for C. However, set of values for B and C are independent of each other.
- MVD between attributes A, B, and C in a relation using the following notation:

  A ——➢➢ B

  A ——➢➢ C

Fourth Normal Form (4NF) : *Multi-Valued Dependency.*

Course	Teachers	Texts
Physics	Prof.Green Prof.Brown	Basic Mechanics. Principles of Optics.
Math	Prof.Green	Basic Mechanics. Vectors. Trigonometry.

- As Shown Above, each tuple consists of a course name, plus a relation containing teacher names plus a relation containing texts names.
- It means that the specified course can be taught by any of the teachers and the teachers can use all of the texts.
- In other words : *Prof.Green* who teaches *physics* can use any of the texts namely *Basic Mechanics* and *Principles of Optics*. And similarly Prof. Brown who teaches physics can also use the 2 texts.

Fourth Normal Form (4NF) : *Multi-Valued Dependency*.

- What that example meant was that there cannot exist a functional dependency COURSE →TEACHER.. Or COURSE →TEXT.
- Instead there are 2 MVD's that hold :
  - COURSE ->> TEACHER and
  - COURSE ->> TEXT.
- i.e. Although a course doesn't have a 'Single' corresponding teacher, nevertheless each Course has a well defined set of corresponding teachers..

Fourth Normal Form (4NF) : *Multi-Valued Dependency*.

- Each tuple of the table above therefore gives rise to *m*n* tuples as shown below.

Course	Teachers	Texts
Physics	Prof.Green Prof.Brown	Basic Mechanics. Principles of Optics.
Math	Prof.Green	Basic Mechanics. Vectors. Trigonometry.

Course	Teachers	Texts
Physics	Prof. Green	Basic Mechanics.
Physics	Prof. Green	Principles of Optics.
Physics	Prof.Brown	Basic Mechanics.
Physics	Prof.Brown	Principles of Optics.
Math	Prof. Green	Basic Mechanics.
Math	Prof. Green	Vectors.
Math	Prof. Green	Trigonometry.

Fourth Normal Form (4NF)

•Definition : Relation R is said to be in the fourth normal form if and only if *it is in BCNF* and *all the multi-valued dependencies are also functional dependencies.*

•Example : Consider the same COURSE-TEACHER-TEXT table shown along side.

•In the table, each tuple is the sole Candidate key.

•In order to convert it to fourth normal form, We can split the table as shown below:

Course	Teachers	Texts
Physics	Prof. Green	Basic Mechanics.
Physics	Prof. Green	Principles of Optics.
Physics	Prof.Brown	Basic Mechanics.
Physics	Prof.Brown	Principles of Optics.
Math	Prof. Green	Basic Mechanics.
Math	Prof. Green	Vectors.
Math	Prof. Green	Trigonometry.

Course-Teachers

Course	Teachers
Physics	Prof. Green
Physics	Prof.Brown
Math	Prof. Green

Course-Text

Course	Texts
Physics	Basic Mechanics.
Physics	Principles of Optics.
Math	Basic Mechanics.
Math	Vectors.
Math	Trigonometry.