

# Conceptual Database Design

## Normalization

---

### GIS Applications

Spring 2009

### How should data items be organized to form relations that describe entities and relationships between entities?

---

- Normalization is a foundation for relational database design
- It involves removing redundant data from relational tables by decomposing a relational table into smaller tables.

#### Objectives

- Minimize unnecessary redundancies
- Support general purpose query processing
- Minimize unwanted side effects of database updates
  - Inadvertent deletions or insertion errors

## Normalization

---

- Normalization theory is based on the concepts of **normal forms**.
- A relational table is said to be in a particular **normal form** if it satisfies a certain set of constraints.
- There are six normal forms that have been defined.
- The first three normal forms defined by E. F. Codd are typically required of all tables in a relational database
- The fourth and fifth normal forms are relatively rare
- Based on the analysis of functional dependencies among attributes.

## Functional Dependence

---

- The concept of functional dependence is the basis for the first four normal forms.
- Given a relation R, a column Y, of 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
- 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.
- Such a functional dependency is denoted as  $X \rightarrow Y$

## The issue of keys

The goal of database normalization is to ensure that every non-key column in every table is **directly dependent on the key, the whole key and nothing but the key**.

## Candidate and Primary Keys

**Superkey** – a set of one or more attributes that uniquely identifies a specific instance of an entity

**Candidate key** – any subset of the attributes of a superkey that is also a superkey and not reducible to another superkey

**Primary key** – a selection from the set of candidate keys - used to index a relation

Dog registry: (name, owner, address, phone)

Field test: (field#, year, crop)

Catalog Order: (catalog\_item #, customer #, billing\_address, shipping\_address)

## Primary Keys

Every relation (entity) must have a **primary key**

To qualify as a primary key, an attribute must have the following properties:

- it must have a non-null value for each instance of the entity
- the value must be unique for each instance of an entity
- the values must not change or become null during the life of each entity instance

## Composite Keys

Sometimes more than one attribute is required to uniquely identify an entity. A primary key made up of more than one attribute is known as a *composite key*.

Student #	Student Name	Major
38214	Bright	IS
38214	Bright	EE
69173	Smith	PHY

## Full Functional Dependence

- Applies to tables with composite keys
- Column Y in relational table R is fully functionally 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, then the other columns must be identified by the entire key and not just some of the columns that make up the key.

## Foreign Keys

- A foreign key identifies a column or a set of columns in one (referencing) table that refers to a column or set of columns in another (referenced) table.
- Columns in the referencing table must be the primary key or other candidate key in the referenced table.
- A foreign key completes a relationship by identifying the parent entity.
- Foreign keys provide a method for maintaining integrity in the data (called referential integrity) and for navigating between different instances of an entity.
- Every relationship in the model must be supported by a foreign key.

## Steps in Normalization

- Assemble data items from user views
- Convert to un-normalized relations
- Convert to first normal form (1NF)
- Convert to second normal form (2NF)
- Convert to third normal form (3NF)

Should result in simple relations that correspond to entities or associations between entity classes

Normalized tables when recombined (joined), should convey exactly the same information as the original table.

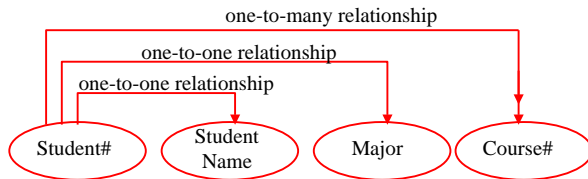
## Un-Normalized Relations

Un-normalized relations contain one or more repeating groups – multiple values at the intersection of rows and columns

Student #	Student Name	Major	Course #	Course title	Instructor name	Instructor Location	Grade
38214	Bright	IS	IS 350	Databases	Codd	B104	A
			IS 465	System Analysis	Kemp	B213	C
69173	Jones	PM	IS 465	System Analysis	Kemp	B213	A
			PM 300	Prod Mang	Lewis	D317	B
			QM 440	Op Res	Kemp	B213	C

Contains redundant information e.g IS 465 appears in more than one row

## Un-Normalized Relations



Need to address the one-to-many relationships

## Normalized relations: First Normal Form

- A relation is in first normal form if the underlying domains contain only atomic values
- There are no repeating groups within a tuple
- Most relational systems require a database to be in 1NF

## First Normal Form

Remove repeating groups and form 2 new relations – migrate the primary key, and assure there is a valid new primary key

<u>Student #</u>	Student Name	Major
38214	Bright	IS
69173	Smith	PM

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

## Identification of Primary Key

<u>Student #</u>	Student Name	Major
38214	Bright	IS
69173	Smith	PM

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

## Insert anomaly

Insertion of a new course cannot occur until a student has registered for the course since Student # is part of the composite key

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

## Update anomaly

Changing a course title or course number requires searching all tuples to find every occurrence of a course number or title

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

## Deletion anomaly

Dropping a single student from a course requires dropping the course and losing the associated course and instructor information

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

## Functional Dependencies

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

Course #  $\rightarrow$  Course Title

Course #  $\rightarrow$  Instructor Name

Course #  $\rightarrow$  Instructor Location

Student #, Course #  $\rightarrow$  Grade

## Second Normal Form

A relation is in second normal form if it is in 1NF and every non-key attribute is fully dependent on the primary key

<u>Student #</u>	<u>Course #</u>	Course Title	Instructor name	Instructor Location	Grade
38214	IS 350	Database	Codd	B104	A
38214	IS 465	Sys Anal	Kemp	B213	C
69173	IS 465	Sys Anal	Kemp	B213	A
69173	PM 300	Op Res	Lewis	D317	B

Course Title, Instructor Name and Instructor Location are **partially dependent** (only on Course#) on the primary key

## Second Normal Form

To convert from first to second normal form – remove partial dependencies

Create 2 new relations, one with attributes fully dependent on primary key, other with attributes only partially dependent

<u>Student #</u>	<u>Course #</u>	Grade
38214	IS 350	A
38214	IS 465	C
69173	IS 465	B
69173	PM 300	C

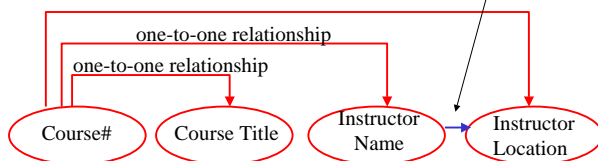
Courses are independent of Student # and so can be inserted or deleted independently, only a single tuple needs to be updated in the course relation

<u>Course #</u>	Course Title	Instructor Name	Instructor Location
IS 350	Database	Codd	B104
IS 465	Sys Anal	Kemp	B213
PM 300	Prod man	Lewis	D317
QM 440	Op Res	Kemp	B213

## Transitive dependencies

<u>Course #</u>	Course Title	Instructor Name	Instructor Location
IS 350	Database	Codd	B104
IS 465	Sys Anal	Kemp	B213
PM 300	Prod man	Lewis	D317
QM 440	Op Res	Kemp	B213

A non-key attribute is dependent on one or more non-key attributes



## Insertion anomaly

Since instructor is dependent on Course # as primary key no information about an instructor can be added until an instructor has been assigned to a course

## Delete anomaly

Deleting data for a course results in deleting instructor information

## Update anomaly

Course #	Course Title	Instructor Name	Instructor Location
IS 350	Database	Codd	B104
IS 465	Sys Anal	Kemp	B213
PM 300	Prod man	Lewis	D317
QM 440	Op Res	Kemp	B213

To update instructor information the entire relation must be searched since instructor information occurs more than once.

## Third Normal Form

- A relation is in third normal form if it is in 2NF and contains no transitive dependencies
- Every non-key attribute is fully dependent on the primary key and there are no transitive dependencies

Instructor Name	Instructor Location
Codd	B104
Kemp	B213
Lewis	D317

Non-key attributes that participate in the transitive dependency form a new relations

Course #	Course title	Instructor Name
IS 350	Database	Codd
IS 465	Sys Anal	Kemp
PM 300	Prod Mang	Lewis
QM 440	OP Res	Kemp

Foreign key – a non-key attribute in one relation that serves as a primary key in another relation

## Boyce-Codd Normal Form

Occurs in the case of overlapping candidate keys

- Each student can major in several subjects
- For each major a student has one advisor
- Each major has several advisors
- Each advisor advises only one major

Student #	Major	Advisor
123	Physics	Einstein
123	Music	Mozart
456	Biol	Darwin
789	Physics	Bohr
999	Physics	Einstein

There are 2 possible candidate keys: Student #-Major or Major -Advisor and they are overlapping.

Attributes that are part of a candidate key are dependent on part of another candidate key.

## Boyce-Codd Normal Form

A relation is in Boyce-Codd normal form if it is in 3NF and there are no dependencies in candidate keys

Student #	Major	Advisor
123	Physics	Einstein
123	Music	Mozart
456	Biol	Darwin
789	Physics	Bohr
999	Physics	Einstein

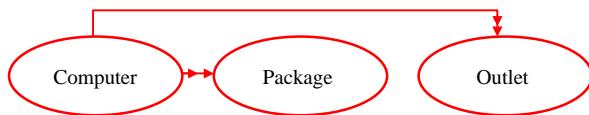
Need to project into 2 new relations

Student #	Advisor	Advisor	Major
123	Einstein	Einstein	Physics
123	Mozart	Mozart	Music
456	Darwin	Darwin	Biol
789	Bohr	Bohr	Physics
999	Einstein		

## Fourth Normal Form

Removes multi-valued dependencies

**Multi-valued dependency** – when 3 attributes (A, B, C) exist in a relation and for each value of A there is a well defined set of values for B and a well defined set of values for C, yet B and C are independent of each other



A record type should not contain two or more independent multi-valued facts about an entity.

## Fourth Normal Form

Computer	Package	Outlet
Apple	Visicalc	Computerland
Apple	Applestar	Computerland
Apple	Visicalc	Byte Shop
Zenith	Wordstar	Computershop
Zenith	Supercalc	Computershop
Zenith	Wordstar	Byte Shop

Several redundancies exist in the relation

Can generate deletion and update anomalies

Computer	Package	Computer	Outlet
Apple	Visicalc	Apple	Computerland
Apple	Applestar	Apple	Byte Shop
Zenith	Wordstar	Zenith	Computershop
Zenith	Supercalc	Zenith	Byte Shop

Project to 2 new relations

## Normalization Summary

Leads to simpler (to implement) applications and to more maintainable systems

Based on a set of rules that define normal forms – of which first three are most important:

**First normal form:** All column values are atomic

**Second normal form:** All column values depend on the value of the primary key: no partial dependencies

**Third normal form:** No column value depends on the value of any other column except the primary key – no transitive dependencies

## Limits of Normalization

- Normalization rules are guidelines
- In certain circumstances 3NF or higher may not be desirable

Customer (Name, Street, City, State, Zipcode)

Does not meet 3NF

Customer(Name, Street, Zipcode)

Location(Zipcode, City, State)

3NF may not be efficient in terms of regular queries

Need to apply judgment and common sense

## Limits of Normalization

---

- There can be a number of cases where there is a compelling need for non first normal form structures.
  
- Spatial data objects is one of them
  
- Object-relational model supports ability to implement non-first normal structures
  - arrays
  - nested tables

## References

---

1. E.F. Codd, "A Relational Model of Data for Large Shared Data Banks", Comm. ACM 13 (6), June 1970, pp. 377-387. The original paper introducing the relational data model.
2. E.F. Codd, "Normalized Data Base Structure: A Brief Tutorial", ACM SIGFIDET Workshop on Data Description, Access, and Control, Nov. 11-12, 1971, San Diego, California, E.F. Codd and A.L. Dean (eds.). An early tutorial on the relational model and normalization.
3. E.F. Codd, "Further Normalization of the Data Base Relational Model", R. Rustin (ed.), Data Base Systems (Courant Computer Science Symposia 6), Prentice-Hall, 1972. Also IBM Research Report RJ909. The first formal treatment of second and third normal forms.
4. C.J. Date, An Introduction to Database Systems (third edition), Addison-Wesley, 1981.