DATABASE SYSTEMS

CS – 355/ CE – 373

Instructor: Maria N. Samad

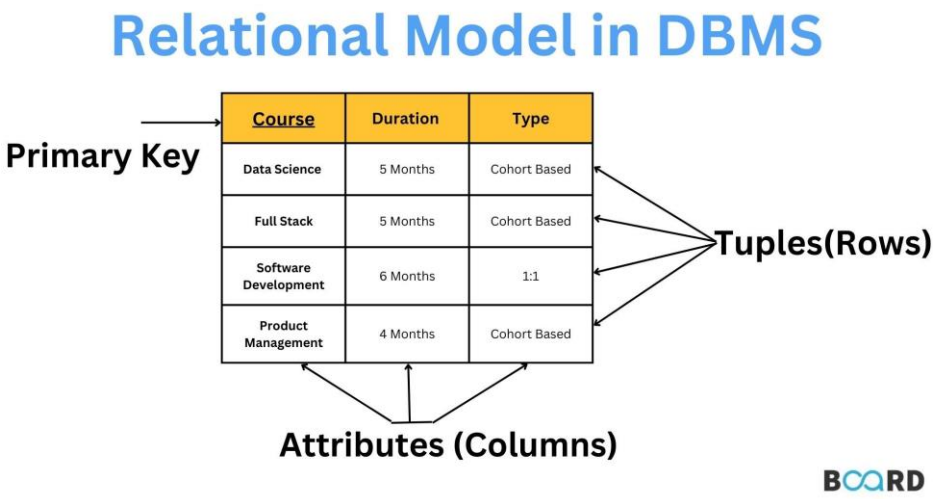
August 28th, 2024

DATA MODEL

- A collection of conceptual tools to describe:
  - ***Data***
  - ***Data Relationships***
  - ***Data Semantics/Rules***
  - ***Consistency Constraints***
- Provides a way to describe the overall design of a database at all the levels of abstraction
- The 4 main categories of data models are:
  - Relational Model
  - Entity-Relationship Model
  - Object-Based Data Model
  - Semistructured Data Model

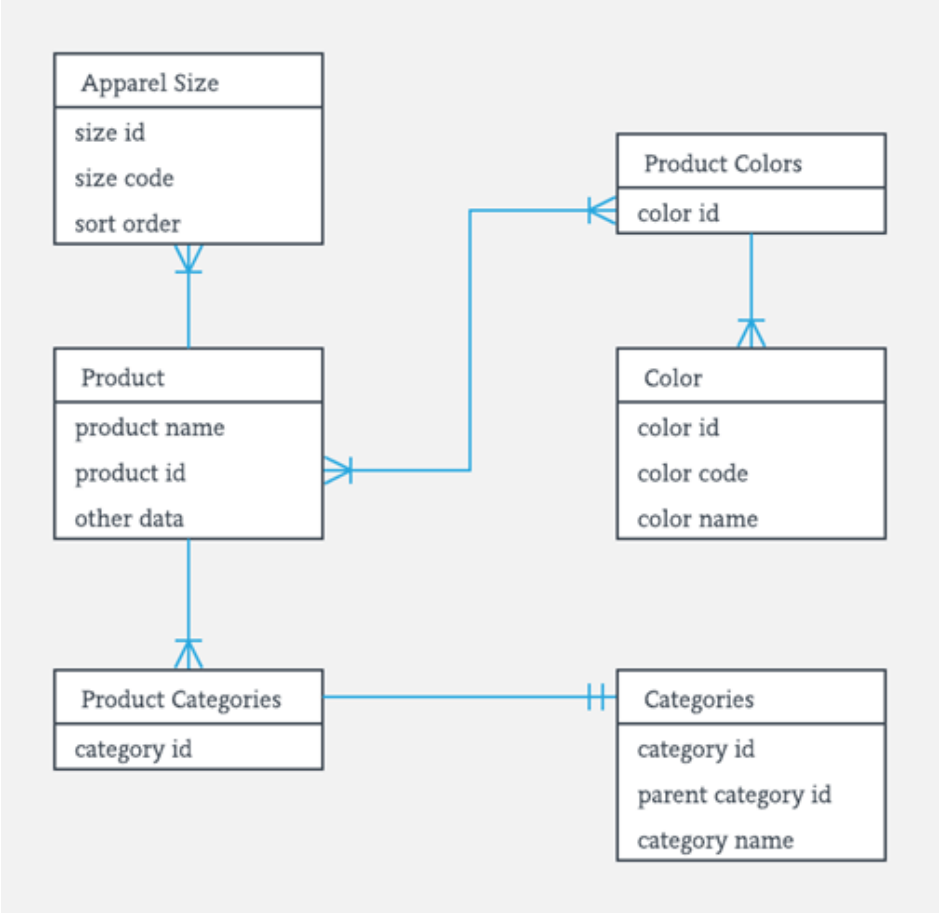
DATA MODEL – RELATIONAL MODEL

- Uses a collection of tables to represent both data and their relationships
- Tables are also known as ***relations***
- Based on ***record-based model***
- Each table has ***multiple columns***, and each column must have a ***unique name***
  - It has to be unique for that particular relation only, and not for the entire database
- Each row is called a ***tuple*** or ***record***, which represents a particular entity or object
- Each record has a ***fixed number of fields or attributes***, which are represented by the ***columns***
- Most widely used data model



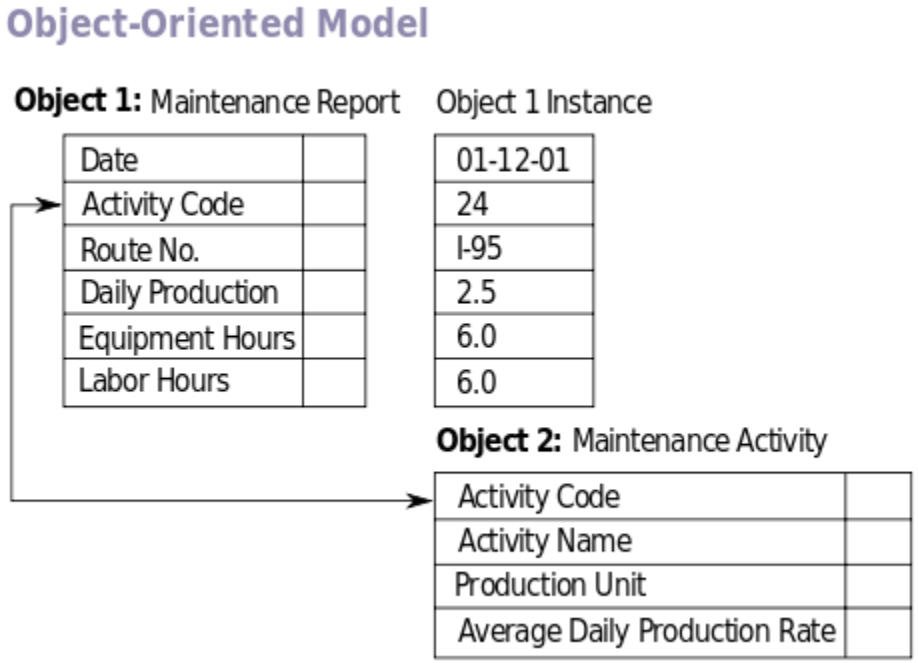
DATA MODEL – ENTITY-RELATIONSHIP MODEL

- Uses a collection of basic objects called ***entities*** and their ***relationships*** among each other
- An ***entity*** can represent any thing or object of the real world
- Depicted by E-R Diagrams



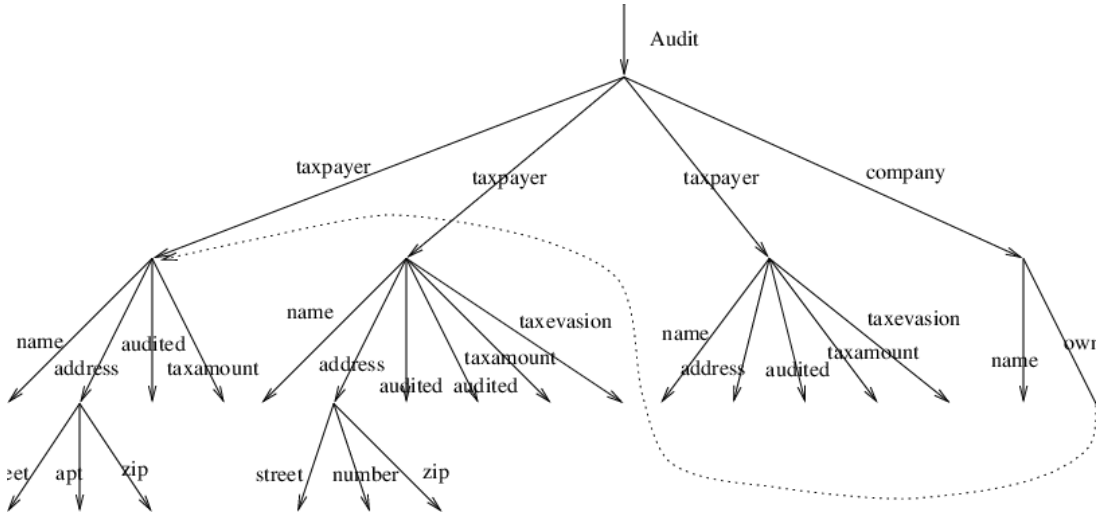
DATA MODEL – OBJECT-BASED DATA MODEL

- An extension of E-R Model
- Includes Object Oriented Programming relationships as well, such as encapsulation, methods, object identities, etc



DATA MODEL – SEMISTRUCTURED DATA MODEL

- A type of data that is not purely structured, but also not completely unstructured.
- It contains some level of organization or structure, but does not conform to a rigid schema or data model, and may contain elements that are not easily categorized or classified
- Use another structures like graphs or trees to represent the data

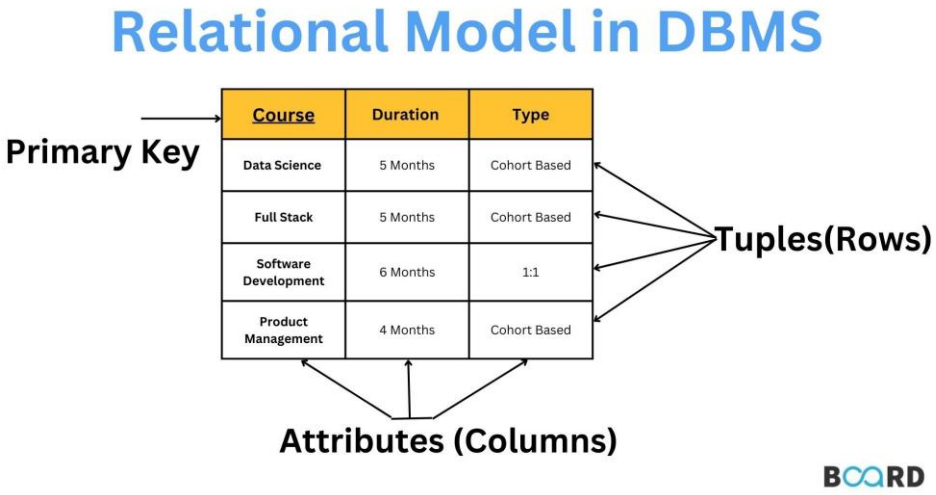


DATA MODEL – OTHERS

- Historically, ***Network Data Model*** and ***Hierarchical Data Model*** preceded Relational Models
- Provided very less or none program-data independence
- Not used much now, except for in old database code that may be still in service in some places

DATA MODEL – RELATIONAL MODEL

- Uses a collection of tables to represent both data and their relationships
- Tables are also known as ***relations***, and each must have a unique name
- Based on ***record-based model***
- Each table has ***multiple columns***, and each column must also have a ***unique name***
- Each row represents a particular entity or object
- Each record has a ***fixed number of fields or attributes***, which are represented by the ***columns***
- Most widely used data model



RELATIONAL DATA MODEL - TERMINOLOGY

- **Tuple:**
  - Each row in a table *records information*, and *represents a relationship* among a set of values within that table
  - A single row is technically known as ***tuple*** or ***record***
  - Each attribute in a tuple is related to each other
  - For example, in the given Instructor table, each row or ***tuple*** is representing information about any one specific instructor; and all the attributes are related to each other in such a way that they represent one particular feature of that instructor, which collectively will represent one instructor's information

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Figure 2.1 The *instructor* relation.

RELATIONAL DATA MODEL - TERMINOLOGY

- **Attribute:**
  - Each column in the table represents one feature or characteristic of that table, and is known as the ***attribute***
  - All the attributes collectively will represent one record of a table
  - For example, in the given Instructor table, ***ID*** represents one of the attributes of the table
  - Similarly, ***Name***, ***Dept_Name***, and ***Salary*** are all different attributes of the table
  - Using ***ID***, ***Name***, ***Dept_Name***, and ***Salary***, you can access any one complete record

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Figure 2.1 The *instructor* relation.

RELATIONAL DATA MODEL - TERMINOLOGY

- **<u>Relation:</u>**
  - Each table in the Relational Data Model is known as ***relation***
  - Since each row represents a relationship among a set of values from each column, so a table is considered as a ***collection of relationships***, as there are more than one rows in the table
  - As each row is called a ***tuple*** with certain number of ***attributes***, let's say **n** attributes, hence a single record in the relation is said to be an ***n-tuple values***

RELATIONAL DATA MODEL - TERMINOLOGY

- **Relational Instance:**
  - Refers to a specific instance of a table at any given time with some defined values in each row and column
  - The order in which tuples appear in ***relational instance*** does not matter, as it is just a set of tuples, and not necessarily ordered/sorted set

RELATIONAL DATA MODEL - TERMINOLOGY

- **Domain:**
  - Each attribute has a set of permitted values, which are called the ***domain*** of that attribute
  - A domain must be ***atomic*** which means they must be singular, indivisible values
  - The ***null*** value signifies that the attribute value is either unknown or does not exist
  - Null values can cause a number of problems while accessing data, so they should be eliminated, or at least handled when querying the database
  - Even if the null values are being handled during querying process, key attribute can **NEVER** have a null value

RELATIONAL DATA MODEL - EXAMPLE

Relation

Table

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Attribute:
dept_name

Tuple:
(83821, Brandt,
Comp. Sci., 92000)

Figure 2.1 The *instructor* relation.

RELATIONAL DATA MODEL - EXAMPLE

- Identify some **attributes** in the course relation in the given figure.
- Identify any **tuple** in the relation.
- For each attribute in the relation, are there any attributes that have **unique** values?
- Are there any attributes with **non-unique** values?

course_id	*title*	*dept_name*	*credits*
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

Figure 2.2 The *course* relation.

EXAMPLES

- Suppose that we add an attribute **Phone Number** to the instructor relation.
- Is this attribute **atomic**?
- Can this attribute have a '**null**' value?

ID	*name*	*dept_name*	*salary*	Phone Number
10101	Srinivasan	Comp. Sci.	65000	
12121	Wu	Finance	90000	1234, 5259
15151	Mozart	Music	40000	
22222	Einstein	Physics	95000	
32343	El Said	History	60000	
33456	Gold	Physics	87000	7985
45565	Katz	Comp. Sci.	75000	
58583	Califieri	History	62000	*null*
76543	Singh	Finance	80000	
76766	Crick	Biology	72000	
83821	Brandt	Comp. Sci.	92000	
98345	Kim	Elec. Eng.	80000	

INSTANCES AND SCHEMAS

- ***Instance of the Database:***
  - The collection of information stored in Database at any particular moment in time
- ***Database schema***
  - The overall design of the database
  - Schemas rarely change, as they have already been planned and designed at the Database design stage
- Example:
  - Database schema would correspond to different data fields and/or their types and constraints
  - The values of these data items at any point in time, which can change later corresponds to the instance of the database

INSTANCES AND SCHEMAS

- Schemas are represented as relations, and modeled using Schema Diagrams
- ***Relational Schema:***
  - Specify the name of the relation, as well as its attributes and sometimes indicating the primary keys as underlined attributes
  - When mentioning the primary key, it should always be placed first in the list of attributes
  - For example, Instructor relation will have the following schema:
    - ***Instructor(<u>ID</u>, Name, Dept_Name, Salary)***
  - And Course relation schema will be:
    - ***Course(<u>CourseID</u>, Title, Dept_Name, Credits)***

LINKING RELATIONS

- Consider the department schema (Fig 2.5), and the instructor schema (Fig 2.1)

  department(*dept_name*, *building*, *budget*) ,
  instructor(*ID*, *name*, *dept_name*, *salary*)

- Observe that the attribute *dept_name* is duplicated in both.
- This duplication is useful in answering queries which involve multiple relations.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Figure 2.1 The *instructor* relation.

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Figure 2.5 The *department* relation.

QUERY EXAMPLE

- Find the names of **all** instructors who work in the Watson building?

dept_name	*building*	*budget*
Biology	Watson	90000
Comp. Sci.	Taylor	100000
Elec. Eng.	Taylor	85000
Finance	Painter	120000
History	Painter	50000
Music	Packard	80000
Physics	Watson	70000

Figure 2.5 The *department* relation.

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Figure 2.1 The *instructor* relation.

KEYS

- There must a be a way to specify how tuples within a given relation are distinguished from each other
- This is done by having different/unique values of their attributes
- The values of attributes of a tuple must be defined in a way that they can ***uniquely identify*** a record
- This means that no two tuples in a relation will have the same values for all their attributes
- The attribute that uniquely identifies the tuple is known as the ***key***

KEYS

- There are different types of keys:
  - Superkey
  - Candidate Key
  - Primary Key
  - Foreign Key
  - Surrogate Key

SUPERKEY

- A set of single attributes, or combination of attributes that uniquely identify a tuple in a relation is known as ***superkey(s)***
- For example, in the Instructor relation, we can have the following combinations of attributes:
  - {ID}
  - {Name}
  - {Dept_Name}
  - {Salary}
  - {ID, Name}
  - {ID, Dept_Name}
  - {ID, Salary}
  - {Name, Dept_Name}
  - {Name, Salary}
  - {Dept_Name, Salary}
  - {ID, Name, Dept_Name}
  - {ID, Name, Salary}
  - {ID, Dept_Name, Salary}
  - {Name, Dept_Name, Salary}
  - {ID, Name, Dept_Name, Salary}

ID	*name*	*dept_name*	*salary*
10101	Srinivasan	Comp. Sci.	65000
12121	Wu	Finance	90000
15151	Mozart	Music	40000
22222	Einstein	Physics	95000
32343	El Said	History	60000
33456	Gold	Physics	87000
45565	Katz	Comp. Sci.	75000
58583	Califieri	History	62000
76543	Singh	Finance	80000
76766	Crick	Biology	72000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000

Figure 2.1 The *instructor* relation.

SUPERKEY

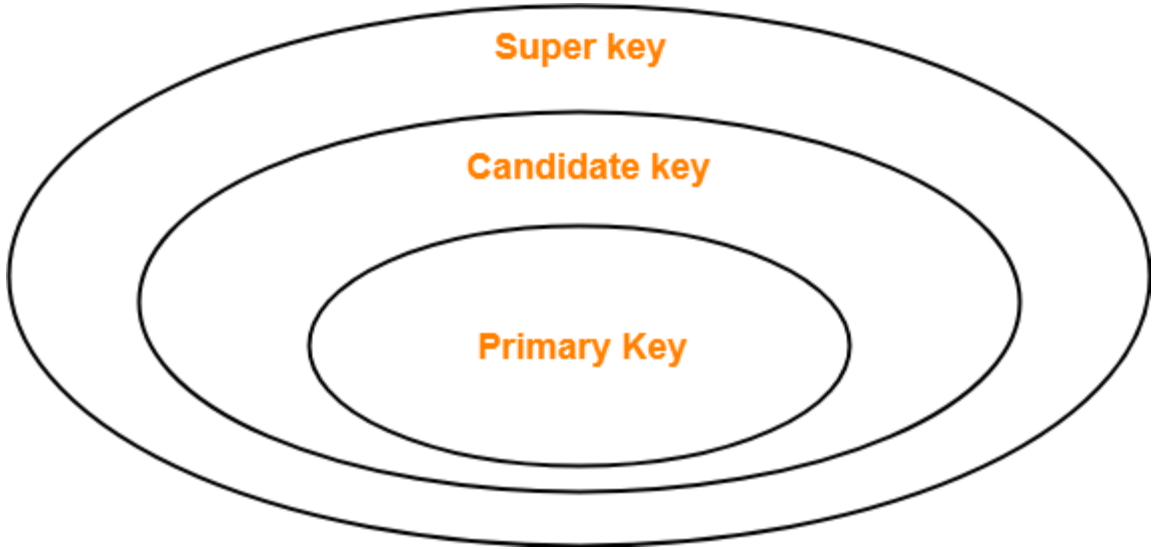
- First thing that must be checked if any of these combination of attributes can be defined as a ***superkey*** or not, i.e. to check if they will be able to uniquely identify a tuple in a relation
- In the Instructor relation, the keys should give some information about the instructor, and that too, uniquely.
- The ones that will work include:
  - {ID}
  - {ID, Name}
  - {ID, Dept_Name}
  - {ID, Salary}
  - {ID, Name, Dept_Name}
  - {ID, Name, Salary}
  - {ID, Dept_Name, Salary}
  - {ID, Name, Dept_Name, Salary}
- All other subsets are discarded, as their values can be repeated and may not be unique, so only the ones shown above form a set of superkeys

CANDIDATE KEY

- Having so many subsets may complicate the overall querying process, so the number must be reduced
- The minimal set of superkeys are known as ***candidate keys***
- The minimal set is defined as single superkey or set of superkeys for which no further subset is a superkey
- In the previous example, all the superkeys, except for ***{ID}*** will have {ID} as their subset which is already a superkey, so we discard all the other superkeys, and use just ***{ID}*** as our candidate key
- If a set, let's say ***{A, B}*** forms a superkey, but neither ***{A}*** nor ***{B}*** is a superkey, then ***{A, B}*** will be considered as the minimal set, so chosen as a candidate key as well

PRIMARY KEY

- One or more than one candidate keys chosen by the database designer as the principal means of identifying tuples in a relation is known as the ***primary key***



FOREIGN KEY

- A primary key attribute of one relation, say R1 is also present in the relational schema of another relation, say R2, it is known as the ***foreign key***
- Here, R2 is called the ***referencing relation***
- And R1 is called the ***referenced relation***
- For example, consider the following relational schemas:
  - ***department*(*dept_name*, *building*, *budget*)**
  - ***instructor*(*ID*, *name*, *dept_name*, *salary*)**
- The primary key of department, i.e. dept_name is a foreign key in instructor
- Foreign Key is defined from **Instructor** to **Department**, but not vice versa

SURROGATE KEY

- A kind of primary key that is not derived from the given data, nor have a business generated values, like studentID, CNIC, CourseID, etc
- This is generated by the Database Systems
- Usually a sequence number or index number
- Independent of all other attributes, i.e. has no actual relationship with the rest of the data
- Used mostly in Agile systems, where tight feedback cycles and continuous changes of data do not allow any particular attribute to remain unique

DATABASE SCHEMA – USING KEYS

- Activity Sheet

DATABASE SCHEMA – USING KEYS

- Activity Sheet Solution:
  - Keys Solution