بسم الله الرحمن الرحيم

In the Name of

Allah

The Most Merciful and Compassionate the most gracious and beneficent, Whose help and guidance we always solicit at every step, at every moment.

Database System Concepts

Chapter No 05
Relational Database Design

SADULLAH KARIMI

Introduction

- The relational data model represents data in the form of tables.
- The relational data model consists of the following three components:
- Data structure
- Data Manipulation
- Data integrity

Introduction

- **Data structure**
- Data are organized in the form of tables with rows and columns.
- **Data Manipulation**
- Powerful operations (using the SQL language) are used to manipulate(control) data stored in the relations.
- **Data integrity**
- Facilities are included to specify business rules that maintain the integrity of data when they are manipulated.

Relational Data Structure

- **Relation**
- It is a named, two-dimensional table of data. Each relation (or table) consists of a set of named columns and a random number of unnamed rows.
- **Attribute**
- It is a named column of a relation
- **Row**
- Each row of a relation corresponds to a record that contains data (attribute) values for a single entity. Figure on the next slide shows an example of a relation named EMPLOYEE 1.

Relational Data Structure

Employee1			
Emp_ID	**Name**	**Dept_Name**	**Salary**
100	Margret Sympson	Marketing	48,000
140	Allen Beeton	Accounting	52,000
110	Chris Lucero	Info System	43,000
190	Leonardo Davis	Finance	55,000
150	Susan Martin	Marketing	42,000

This relation contains the following attributes describing employees: Emp_ID, Name, Dept_Name, and Salary. The five rows of the table correspond to five employees.

Relational Data Structure

- It is important to understand that the sample data in Figure in previous slide are intended to illustrate the structure of the EMPLOYEE1 relation; They are not part of the relation itself.
- Even if we add another row of data to the figure, it is still the same EMPLOYEE1 relation.
- Nor does deleting a row change the relation. In fact, we could delete all of the rows shown in Figure, and the EMPLOYEE 1 relation would still exist.

Relational Data Structure

- We can express the structure of a relation by a shorthand notation in which the name of the relation is followed (in parentheses) by the names of the attributes in that relation.
- For EMPLOYEE1 we would have:
- **EMPLOYEE 1 (Emp_ID, Name, Dept_Name, Salary)**

Relational Keys

- We must be able to store and retrieve a row of data in a relation, based on the data values stored in that row.
- To achieve this goal, every relation must have keys.
- **key**
- Attributes are used to differentiate one entity type from another entity type
- An entity type can have many instances; each instance has a value against each attribute.

Keys

- A key is a set of attributes that can be used to identify or access a particular entity instance of an entity type (or out of an entity set).
- The concept of key is beautiful and very useful.
- **why and how?**
- An entity type may have many instances, from a few to several thousands and even more.
- Now out of many instances, when and if we want to pick a particular/single instance, and many times we do need it, then key is the solution.

Keys

- **For example**
- Think of whole population of Afghanistan, the data of all Afghans lying at one place, say with Tazkira Department people.
- Now if at some time we need to identify a particular person out of all this data, how can we do that?
- Can we use name for that, well think of any name, like Muhammad Aslam Khan, now we may find many people with this name in Pakistan.
- Another option is the combination of name and father name, then again, Muhammad Shoaib s/o Muhammad Karim, there could be so many such pairs. There could be many such examples.

Keys

- For example
- However, if you think about National ID Card number, then no matter whatever is the population of Afghnistan, you will always be able to pick precisely a single person. That is the key.
- While defining an entity type we also generally define the key of that entity type.
- How do we select the key?
- From the study of the real-world system; key attribute(s) already exist there, sometimes they don’t then the designer has to define one.
- A key can be simple, that is, consisting of single attribute, or it could be composite which consists of two or more attributes.

Types of Keys

- Following are the major types of keys:
- Candidate Key
- Primary Key
- Composite Primary Key or Composite Key
- Alternate Key
- Secondary Key
- Foreign Key
- All the keys are discussed here as follows:

Candidate key

- One Key or more keys that are candidate for becoming the primary key Is called Candidate Keys

Candidate Key

Candidate Key

Candidate Key

STUDENT

Reg_No	CNIC	Name	F.Name	Address	Class_Name	Admn_Date
0001-10	122011111	Ajmal Zeb	M.Zeb	Kabul	BBA	10-10-2010
0029-11	122011112	Nisar Ahmad	Wisal	Kabul	BCS	21-03-2011
0135-09	122011113	Abid Hussain	M.Ali	kabul	BIT	18-03-2009
0099-10	122011114	Zarka	Jawed	Paktiya	Law	11-05-2011
1001-10	122011115	Ali Rehman	M.Rehman	Wardag	Political Science	30-06-2010
1000-11	122011116	Freshta	Ahmad	Panjsher	BCS	22-07-2011

Primary key

- A candidate key chosen by the database designer to act as key is the primary key. It is an attribute (or combination of attributes) that uniquely identifies each row in a relation.
- An entity type may have more than one candidate keys, in that case the database designer has to designate one of them as primary key.
- Since there is always only a single primary key in an entity type. If there is just one candidate key then obviously the same will be declared as primary key.
- The primary key can also be defined as the **successful candidate k**ey.

Primary key

- Figure below contains the entity type **STUDENT** of previous figure but with an additional attribute **nIdNumber** **(national ID card Number).**
- In figure on next slide , we can identify two different attributes that can individually identify the entity instances of **STUDENT** and they are Reg_No and nIdNumber.
- Both are minimal super keys so both are candidate keys.

Primary key

nIdNumber	Name	Reg_No	Fname	address
10001	Ajmal Zeb	0001-10	M.Zeb	Kabul
20003	Nisar Ahmad	0029-11	Wisal	Kabul
50006	Abid Hussain	0135-09	M.Ali	kabul
90033	Zarka	0099-10	Jawed	Paktiya
10345	Ali Rehman	1001-10	M.Rehman	Wardag
43560	Freshta	1000-11	Ahmad	Panjsher

Both are minimal super keys or both are candidate keys

Primary key

- Now in this situation we have got two **candidate keys**.
- The one that we choose will be declared as **primary key**
- Other will be the **alternate key.**
- Any of the **candidate keys** can be selected as **primary key**, it mainly depends on the database designer which choice he/she makes.

Primary key

Primary Key

Alternate Key

nIdNumber	Name	Reg_No	Fname	address
10001	Ajmal Zeb	0001-10	M.Zeb	Kabul
20003	Nisar Ahmad	0029-11	Wisal	Kabul
50006	Abid Hussain	0135-09	M.Ali	kabul
90033	Zarka	0099-10	Jawed	Paktiya
10345	Ali Rehman	1001-10	M.Rehman	Wardag
43560	Freshta	1000-11	Ahmad	Panjsher

Both are minimal super keys or both are candidate keys

Primary key

- There are certain things that are generally considered while making this decision that which **candidate key** should be **primary key**
- Like the **candidate key** that is shorter, easier to remember, to type and is more meaningful is selected as **primary key**.
- These are general recommendations in this regard, but finally it is the decision of the designer and he/she may have his/her own reasons for a particular selection that may be entirely different from those mentioned above.

Primary key

- A major characteristic of the **PK** is that it cannot have the NULL value.
- If **PK** is a composite, then none of the attributes included in the **PK** can have the NULL

Primary key

- **For example**
- If we are using **“name, phone no”** as **PK** of entity type **STUDENT**, then none of the instances may have **NULL** value in either of the **name** or **phone no** or both.
- The concept of a primary key is related to the term identifier defined in previous Chapter.

Composite Primary Key or Composite key

- It is a primary key that consists of more than one attribute.
- **For example**
- The primary key for a relation **DEPENDENT** would likely consist of the combination **Emp_ID** and **Dependent_Name**.
- **EMPLOYEE1 (Emp ID,Name,Dept_Name,Salary)**

Alternate Keys

- Candidate keys which are not chosen as the primary key are known as alternate keys.
- **For example**
- We have two candidate keys of **STUDENT** in figure of previous slide no 79, **Reg_No and nIdNumber**,
- If we select **Reg_No** as **PK** then the **nIdNumber** will be **alternate** key and vice versa.

Secondary Key

- Many times we need to access certain instances of an entity type using the value(s) of **one or more attributes** other than the **PK**.

- The difference in accessing instances using the value of a key or non-key attribute is that the search on the value of PK will always return a single instance (if it exists), where as uniqueness is not guaranteed in case of non-key attribute.

- Such attributes on which we need to access the instances of an entity type that may not necessarily return unique instance is called the secondary key.

Secondary Key

- **For example**
- We want to see how many of our students belong to **Kabul**
- In that case we will access those instances of the STUDENT entity type that contain **“Kabul”** in their address.
- In this case **address** will be called **secondary key**, since we are accessing instances on the basis of its value, and there is no compulsion that we will get a single instance.

Secondary Key

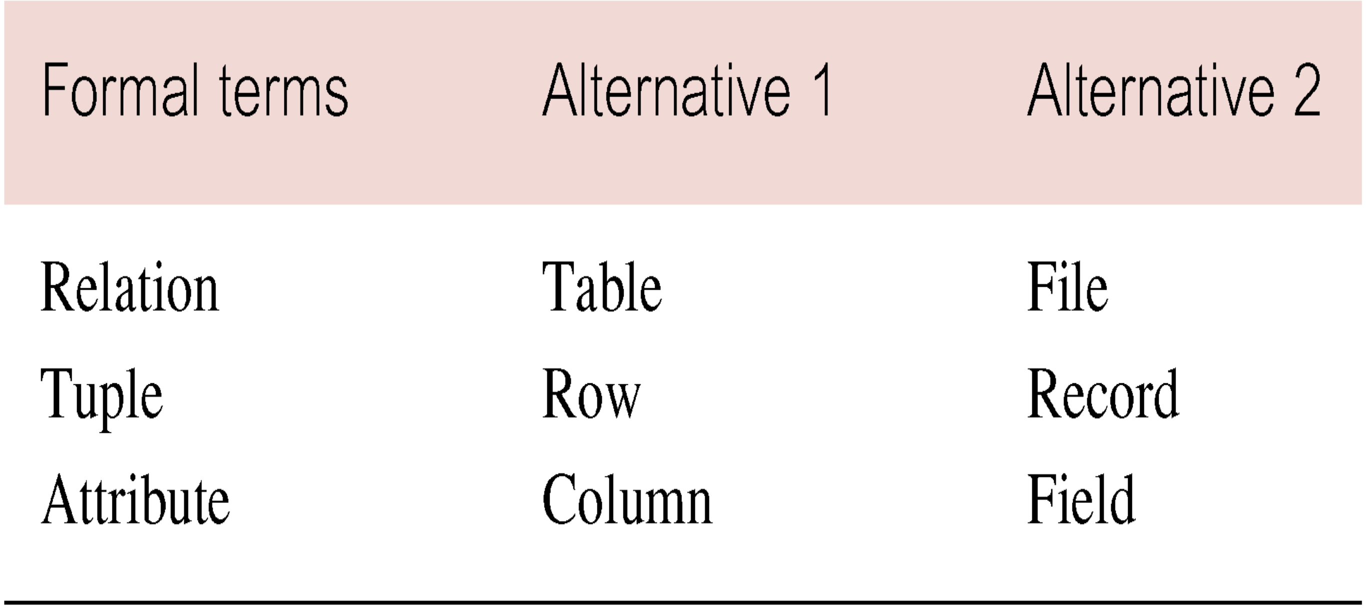
- Keep one thing in mind here, that a particular access on the value of a secondary key MAY return a **single instance**, but that will be considered as chance or due to that particular state of entity set.

- There is not the compulsion or it is not necessary for **secondary key** to return unique instance, where as in case of **super, candidate, primary and alternate keys** it is compulsion that they will always return unique instance against a particular value.

Foreign key

- It is an attribute (possibly composite) in a relation of a database that serves as the primary key of another relation in the same database.
- **For example**
- Consider the relations **EMPLOYEE1** and **DEPARTMENT:**
- EMPLOYEE1 (<u>Emp ID</u>, Name, <u>Dept_Name</u>,Salary)
- DEPARTMENT(<u>Dept Name, </u>Location, Fax)
- The attribute Dept_Name is a foreign key in EMPLOYEE1.
- It allows a user to associate any employee with the department to which he or she is assigned.
- Some authors emphasize the fact that an attribute is a foreign key by using a dashed underline.

Alternative terminologies for the relational model terms



Formal terms	Alternative 1	Alternative 2
Relation	Table	File
Tuple	Row	Record
Attribute	Column	Field

Thanks

Questions?