

Database system

Theoretical lectures

The third lecture

Assistant teacher Shaymaa Abdulkarem



Relational Integrity Constraints -Entity-Set and Keys

Relational Integrity Constraints

In Database Management Systems, integrity constraints are pre-defined set of rules that are applied on the table fields (columns). Integrity constraints are useful in preventing any accidental damage to the database by an authorized user.

Types of Integrity Constraints

- 1 • Key Constraints (entity constraint)
- 2 • Domain Constraint
- 3 • Referential integrity Constraints

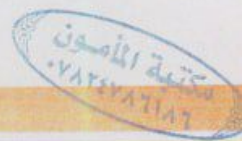
DOMAIN CONSTRAINTS

ID No	Name	Age	DOMAIN
1	Amya	21	
2	Bray	19	
3	John	24	
4	Max	26	

Age must be greater than 18 and must be an integer

ID	NAME	SEMESTER	AGE
1000	Tom	1 st	17
1001	Johnson	2 nd	24
1002	Leonardo	5 th	21
1003	Kate	3 rd	19
1004	Morjan	8 th	A

Not allowed. Because AGE is an integer attribute



Referential integrity Constraints

Referential integrity constraints work on the concept of **Foreign Keys (key = attribute)**.
A foreign key is a key **attribute** of a relation that can be referred in other relation

Primary Table

CompanyId	CompanyName
1	Apple
2	Samsung

Related Table

CompanyId	ProductId	ProductName
1	1	Phone
15	2	Mustang

Associated Record ✓
Orphaned Record ✗

Relational Integrity Constraints

Key Constraints (entity constraint)

There must be at least one minimal subset of attributes in the relation, which can identify a tuple (row) uniquely. This minimal subset of attributes is called key for that relation. If there are more than one such minimal subsets, these are called candidate keys.

There are three key constraints:

- In a relation with a key attribute, no two tuples can have identical values for key attributes.
- A key attribute cannot have **NULL** values. Key constraints are also referred to as Entity Constraints.

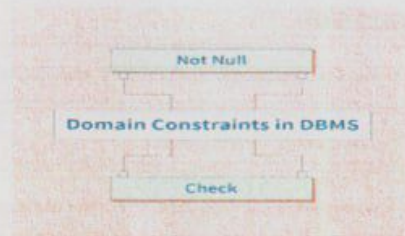
id	name	id	name	Address	id	name	address
1	Ali	1	Ali	Baghdad	1	ali	Baghdad
2	Suha	2	muna	Babel	2	ali	Babel
1	Rami		Sara	Arbil			

Relation invalid

Relation valid

Domain Constraint

- The domain of an attribute specifies all the possible values that attribute can hold.
- Domain constraint defines the domain or set of values for an attribute.
- It specifies that the value taken by the attribute must be the atomic (can't be divided) value from its domain.
- There are 2 constraints which we can study under domain constraint:
 - ✓ Not null constraint
 - ✓ Check clause constraint



Member

Member ID	Name	Date of Birth
1	Alice	03/03/1995
2	Bob	11/07/1993
3	Charlie	21/10/1997
4	Mike	18/09/1992
5	Katie	21/10/1997

Relation Attribute

Tuple

Relational Algebra Operators

$\pi \sigma \rho$

$\bowtie \cup \cap$

Relational Algebra

- Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output
- ✓ It uses operators to perform queries. An operator can be either unary or binary.
- ✓ They accept relations as their input and yield relations as their output.

Relational algebra is performed recursively on a relation and intermediate results are also considered relations. The fundamental operations of relational algebra are as follows: – Select – Project – Union – Set different – Cartesian product – Rename.

Select Operation (σ) It selects tuples that satisfy the given predicate from a relation

Example of Referential Integrity (1)

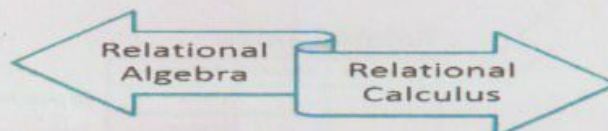
CAR			EMPLOYEE			
RegNo	Type	Owner	ENo	ENAME	M-S	Sal
K123 ABC	Corsa 1.3	E3	E3	Smith	S	12,500
W811 STA	Starlet GLi	E5	E5	Mitchell	M	21,000
JON 1	Jaguar XK	E1	E1	Robson	D	32,500
V771 PQ	Volvo S80	E6	E6	Blake	M	54,000
			E8	Jones	W	68,000

CAR[Owner] is a subset of **EMPLOYEE[ENo]**

Referential integrity constraint states that if a relation refers to a key attribute of a different or same relation, then that key element must exist.

Relational database systems are expected to be equipped with a query language that can assist its users to query the database instances. There are two kinds of query languages:

- ❖ Relational Algebra
- ❖ Relational Calculus



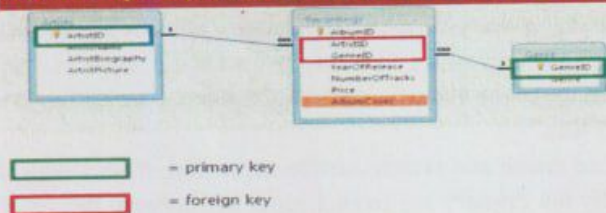
Entity-Set and Keys

Key is an attribute or collection of attributes that uniquely identifies an entity among entity set.

For example, the roll_number of a student makes him/her identifiable among students

- Super Key
- Candidate Key
- Primary Key
- Foreign Key

Primary key, Super key, Candidate Key



Super Key

A set of attributes (one or more) that collectively identifies an entity in an entity set. (Key + other attributes).

Example:

Set of super keys obtained

{ Emp_SSN }
 { Emp_Id }
 { Emp_email }
 { Emp_SSN, Emp_Id }
 { Emp_Id, Emp_name }
 { Emp_SSN, Emp_Id, Emp_email }
 { Emp_SSN, Emp_name, Emp_Id }

Emp_SSN	Emp_Id	Emp_name	Emp_email
11051	01	John	john@email.com
19801	02	Merry	merry@email.com
19801	03	Riddle	riddle@email.com
41201	04	Cary	cary@email.com

Entity

An entity can be a real-world object, either animate or inanimate, that can be easily identifiable.-

For example, in a school database, students, teachers, classes, and courses offered can be considered as entities.

An entity set may contain entities with attribute sharing similar values.-

For example, a Students set may contain all the students of a school; likewise a Teachers set may contain all the teachers of a school from all faculties.

- ✓ Entity sets need not be disjoint.
- ✓ Entities are represented by means of rectangles.
- ✓ Rectangles are named with the entity set they represent.

Student

Teacher

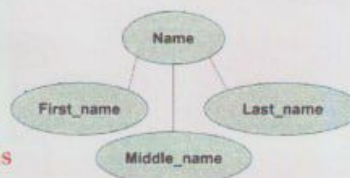
Course

Attributes

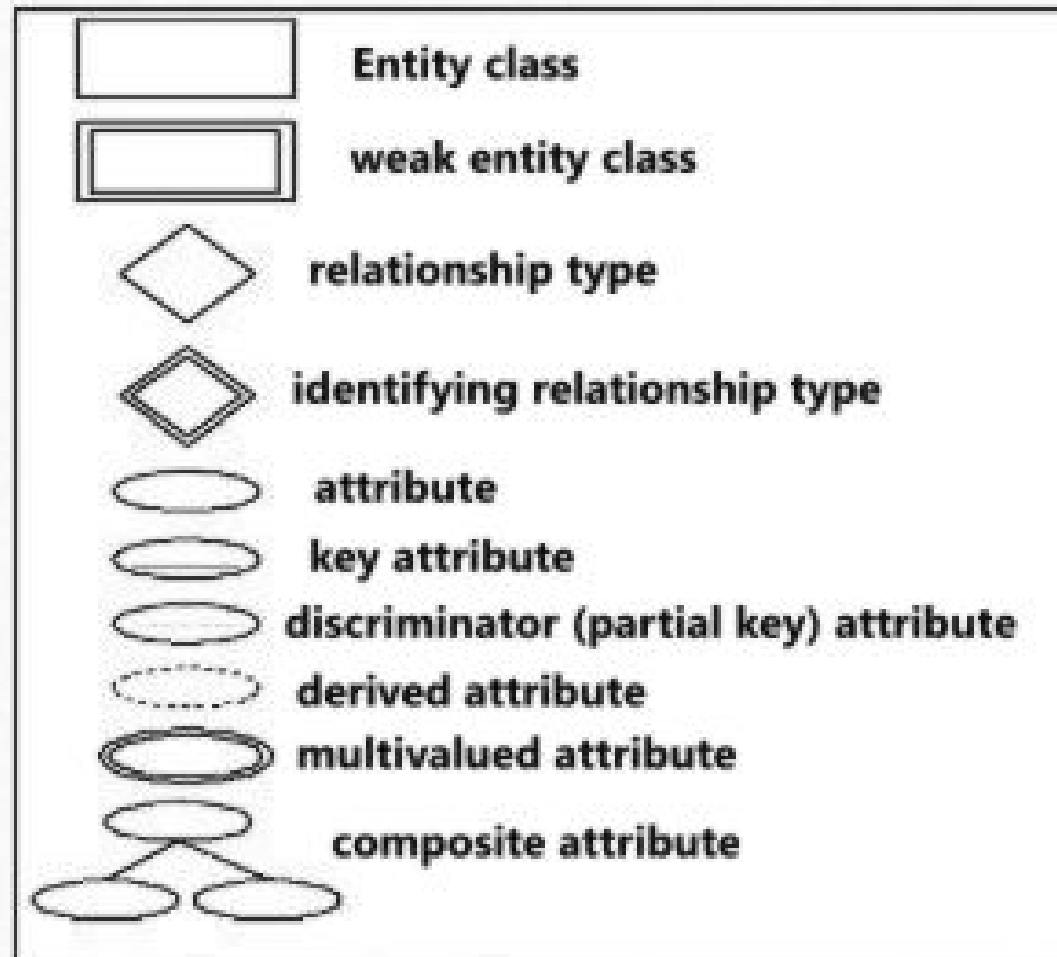
Entities are represented by means of their properties, called attributes. All attributes have values.

- **For example**, a student entity may have name, class, and age as attributes. There exists a domain or range of values that can be assigned to attributes.
- **For example**, a student's name cannot be a numeric value. It must be alphabetic. A student's age cannot be negative, etc.
 - Attributes are the properties of entities.
 - Attributes are represented by means of ellipses.

Every ellipse represents one attribute and is directly connected to its entity (rectangle).



Entity – Attributes -Relationship



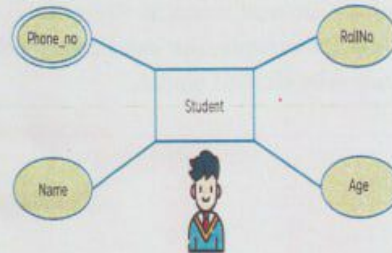
Relationship sets

There are three Relationship:

- ✓ Simple Relationship
- ✓ Composite Relationship
- ✓ Derived Relationship

Simple Relationship : is atomic values, which cannot be divided further.

For example, a student's phone number is an atomic value of 10 digits.



Composite Relationships

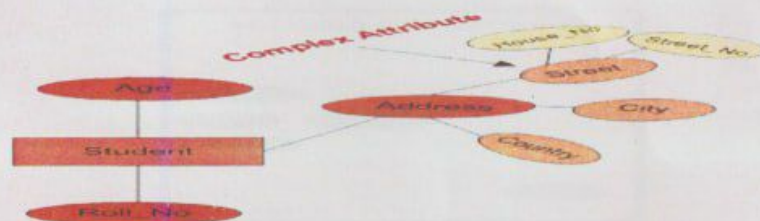
are made of more than one simple attribute.

- For example, a student's address may have country, city and street.

If the attributes are composite, they are further divided in a tree like structure.

Every node is then connected to its Relationship.

That is, composite attributes are represented by ellipses that are connected with an ellipse.



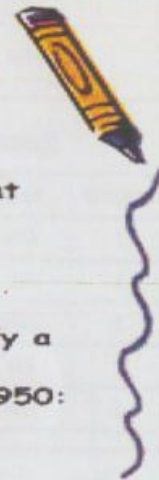
SELECTION (σ)

- The SELECT operator is σ (sigma) symbol
Used as an expression to choose tuples that meet the selection condition...

σ <selection condition>(R)

- > Select operation selects tuples that satisfy a given predicate.

Ex:- find all employees born after 1st Jan 1950:
 σ (employee)



Unary Relational Operations: **PROJECT**

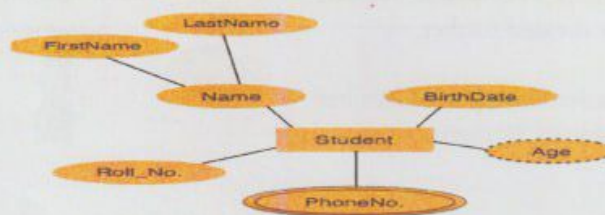
- PROJECT Operation is denoted by π (pi)
- This operation keeps certain *columns* (attributes) from a relation and discards the other columns.
 - **PROJECT creates a vertical partitioning**
 - The list of specified columns (attributes) is kept in each tuple
 - The other attributes in each tuple are discarded
- Example: To list each employee's first and last name and salary, the following is used:

π LNAME, FNAME, SALARY(EMPLOYEE)

Derived Relationship

is the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database.

- For example, average salary in a department should not be saved directly in the database, instead it can be derived.
- Another example, age can be derived from data_of_birth. Derived Relationship is depicted by dashed ellipse.



Single-value Relationship

Single-value Relationship contain single value.

- For example – Social_Security_Number



Multi-value Relationship

may contain more than one values.

- For example, a student can have more than one phone number, email_address, etc.



Relational Calculus

In contrast to Relational Algebra, Relational Calculus is a non-procedural query language, that is, it tells what to do but never explains how to do it.

Relational calculus exists in two forms:

- - Tuple Relational Calculus (TRC)
- - Filtering variable ranges over tuples

Notation - $\{T \mid \text{Condition}\}$

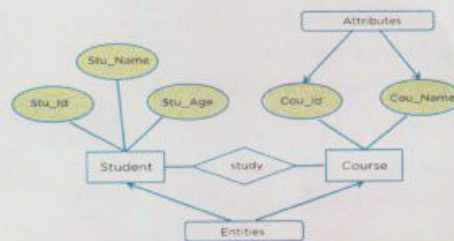
Returns all tuples T that satisfies a condition.

Example : $\{T.\text{name} \mid \text{Author}(T) \text{ AND } T.\text{article} = \text{'database'}\}$ Output

Returns tuples with 'name' from Author who has written article on 'database'

Entity Relationship model

The ER data model was developed to facilitate database design by allowing specification of an enterprise schema that represents the overall logical structure of a database. The ER data model employs three basic concepts: 1. Entity sets, 2. Attributes, 3. Relationship sets. The ER model also has an associated diagrammatic representation, the ER diagram, which can express the overall logical structure of a database graphically.



PrimaryKey:

A foreign key is a column (or combination of columns) in a table whose values must match values of a column in some other table. Its constraints enforce referential integrity, which essentially says that if column value A refers to column value B, then column value B must exist.

Example:

The primary key in the Customers table is **CustomerNo**, and the primary key in the Orders table is **OrderNo**. The primary key uniquely identifies the table and contains the value that the foreign key refers to. Primary keys must be unique to each row of data. In the **Orders** table, the **CustomerNo** value is a foreign key that links the two tables relating back to the **CustomerNo** value in the Customers table.

When a **primary key** in a table changes, the set of associated foreign keys also changes

Customers					Orders				
CustomerNo	FirstName	LastName	OrderNo	EmployeeNo	CustomerNo	Supplier	Price	Item	
1	Sally	Thompson	1	1	42	Hamson	\$215	Desk	
2	Sally	Henderson	2	4	1	Ford	\$234	Chair	
3	Katy	Henderson	3	1	98	Hamson	\$415	Desk	
4	Sandra	Welling	4	2	4	Ford	\$152	Chair	
5	Arthur	Pearson	5	3	6	Ford	\$234	Chair	
7	Don	Beckwith	6	2	112	Ford	\$250	Chair	
8	Ma. De	Jones	7	2	8	Hamson	\$215	Desk	
primary key					foreign key				

Foreign keys are important for several reasons, including the following

- ❖ Streamline data sets: With foreign keys, database administrators don't have to store repeated data in multiple tables. They make data available to different tables without creating redundant data sets. In other words, they act as a cross-referencing system among tables.
- ❖ Promote efficiency: Primary and foreign keys work together to create structure in relational databases, enabling them to sort, search and query data faster.
- ❖ Ensure data integrity: Primary and foreign key relationships also help maintain the data integrity of relational databases. They ensure the existence of a value in the primary table and the accuracy of the foreign key reference in the primary table, even if the primary table is changed or deleted.

▪ Candidate Key:

A minimal super key is called a candidate key. An entity set may have more than one candidate key. (Key- other attributes).

Example:

Candidate Keys :

Emp_SSN
Emp_Id
Emp_email

Emp_SSN	Emp_Id	Emp_name	Emp_email
11051	01	John	john@email.com
19801	02	Merry	merry@email.com
19801	03	Riddle	riddle@email.com
41201	04	Cary	cary@email.com

▪ Primary Key:

A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set. (Key).

One of the Candidate keys.

{Emp-SSN} OR { Emp-id} OR {Emp-email}

Decision made by the database designer.

Emp_SSN	Emp_Id	Emp_name	Emp_email
11051	01	John	john@email.com
19801	02	Merry	merry@email.com
19801	03	Riddle	riddle@email.com
41201	04	Cary	cary@email.com

Role of Super Key

The role of the super key is simply to identify the tuples of the specified table in the database. It is the superset where the candidate key is a part of the super key only. So, all those attributes in a table that is capable of identifying the other attributes of the table in a unique manner are all super keys.

These all are the set of super keys which, together or combining with other prime attributes, can identify a table uniquely.

Just like, if we set Super key on Emp_SSN, it will be able to identify all other tuples of the table very easily. Similarly, if we set the Super key on {Emp_Id, Emp_name}, we can easily get the value or details of the other remaining attributes of the employee. So, in this way,

we can create and search out the super keys from a table. Now, the question is how to identify the Primary key from a super key because the super key is the superset.

So, the answer is that a Primary key is picked up from the super key set only by selecting a single attribute that can provide a unique identification to the whole table. Hence, it is the responsibility of the DBA to choose the most appropriate and suitable primary key out of the candidate keys or super key.

