## MODULE 2

### Domain constraint
A domain constraint violation occurs when an attribute's value does not appear in the corresponding domain.

### Key constraint
Key constraint violation occurs when a tuple is inserted or modified such that the key value is the same as another tuple (i.e. violates the uniqueness constraint of the domain).

### Entity Integrity Constraint
Occurs when a tuple is inserted/modified such that part of the PK contains the value NULL. For PKs that consist of multiple attributes, no part of it can be NULL.

### Referential integrity constraint
When an operation performed on a DB leads to an invalid, incorrect or broken FK reference.
The tuple for which its key is not foreign (i.e. the origin of the FK) exists before any tuples that reference it.

### Semantic integrity constraint
It can be used to enforce organisation policies such as. Implemented in constraint specification language (SQL) using triggers and assertions.

## MODULE 4

## MODULE 3

### Aggregate functions
Functions that produce summary values from a set of tuples.

### Grouping
When you want to consider groups of rows within the table. When GROUP BY is used in an SQL statement, any attribute that appears in the SELECT clause must also appear in the GROUP BY clause or be in an aggregation function.

### Syntax
SELECT [DISTINCT] <target list>
FROM <table list>
[WHERE search condition]
[GROUP BY <grouping attributes> ]
[HAVING <group conditions> ]
[ORDER BY column [ASC|DESC] {, column [ASC|DESC]}];

### Having
The where clause for groups.

### JOIN
SELECT <attribute list>
FROM <table>
{<type of join> JOIN <table to join to>
ON <join attributes>}
[WHERE search condition]

### RECURSIVE JOIN
SELECT A.name AS employee, A.salary AS employeeSalary, B.name AS manager, B.salary AS managerSalary
FROM Employee A
JOIN Employee B ON A.mgrSSN = B.ssn
WHERE A.salary < B.salary

### INNER JOIN
A tuple is included in the result relation only if matching tuples exist in both relations. JOIN defaults to INNER JOIN.

### OUTER JOIN
Left join - includes all rows from the first table
Right join - includes all rows from the right table

## MODULE 4

### Candidate key
A minimal (set of attributes without any redundant attributes) set of attributes that uniquely identify tuples in a relationship. There can be many candidate keys for one relation.
Any superkey must contain at least one candidate key.

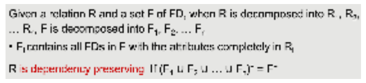
### Primary Key
A primary key is the candidate key that is chosen as the main key for the relation that is underlined in the ERD. One of the selected candidate keys.

### Foreign Key
When mapping relationships to relations, a foreign key may be required. It is a PK from another table. It is a referential key (ie. a key that references a key from another relation).

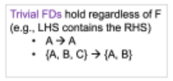
### Superkey
Uniquely identifies the relation. No two tuples can have the same values for these attributes.

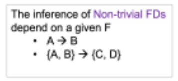
### Prime/non-prime attribute
An attribute in any candidate key (not necessarily a member of the PK). Non-prime – opposite of prime.

### Functional Dependency
An FD X->Y is a relationship where the value of the attribute Y depends on the attribute X in a relation.

### Dependency Preservation

Given a relation R and a set F of FD, when R is decomposed into R, , $R_2$, … R., F is decomposed into $F_1$, $F_2$, … $F_r$
• $F_i$ contains all FDs in F with the attributes completely in $R_i$
R is dependency preserving If $(F_1 \cup F_2 \cup \ldots \cup F_r)^+ = F^+$

Every normalisation process is dependency preserving except for BCNF.

### Closure of F

Trivial FDs hold regardless of F (e.g., LHS contains the RHS)
- A → A
- {A, B, C} → {A, B}


The inference of Non-trivial FDs depend on a given F
- A → B
- {A, B} → {C, D}

### Closure of X
Denoted as X+ is the set of attributes determined by X under F. It is easy to compute.
- X+ initially contains all attributes of X
- For each FD in the set F: if the LHS of the FD is a subset of X+ then add the RHS to X+
- if step 2 resulted in changes in X+ then repeat 2, otherwise finish

### Key Finding
Given a relation R and an associated set of FDs F, for any subset S of attributes in R, S is a key iif (1) S+= R (i.e. the subset determines the relation) and (2) there is no S' ⊂ S such that S+=R (no subset of that subset determines the relation).

### Transitive Dependency
A functional dependency of X->Y in a relation R is a transitive dependency if there exists a set of attributes Z in R such that:
- Z is neither a candidate key nor a subset of any key of R
- Both X -> Z and Z -> Y hold

### Partial Dependency
A FD X->Y such that X is a proper subset of the candidate key.

### 2NF
All FDs are not partial dependencies or the RHS is a prime attribute.

### 0NF
A table.

### 1NF
No non-atomic values.

### Basic set operators
-UNION (UNION [ALL] SELECT ...)
-INTERSECTION (INTERSECT [ALL] SELECT ...)
-DIFFERENCE/MINUS (EXCEPT [ALL] SELECT ...)

### Union compatibility
Where there are the same number of columns
Pair-wise compatible domains

### Set operations and duplicates
Each set operation automatically eliminates duplicates.
To retain duplicates the ALL keyword must be used after the set operation.

### JOIN vs Set Operations
- Combines data from many tables based on a matched condition
- Contains data in new columns
- Number of columns from each table don't have to be the same
- Datatypes of corresponding columns can be different
- It returns duplicate rows by default

- Combines the result of two or more SELECT statements
- Combines the data into new rows
- Number of columns from each table should be the same
- Datatypes from corresponding columns should be the same
- Returns distinct by default

## MODULE 5

### DAC
Grant/revoke privileges. Revoking is done in a chain. Allows owners to grant, and those given grant option to also grant without the owner knowing. Privileges : read/modify/reference tables.

### MAC
Classify data and users into various security classes and implement security policy.
BASIC SECURITY PRINCIPLE - NRU STAR – NWD (No write down)
Typical security classes:
- Top secret (TS)
- Secret (S)
- Confidential (C)
- Unclassified (U)

### ROLE BASED
Roles with preset DAC privileges or MAC security classes.

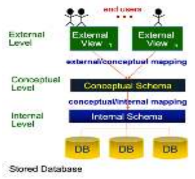
## MODULE 0

### DBMS
Defining, constructing, manipulating, sharing.

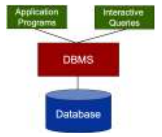
Data integrity maintenance, query processing, security management, concurrency control, backup and recovery.

### Data Independence
Can change the schema at one level without changing it at the next higher level.
Logical Data Independence - the ability to change the conceptual schema without changing external views or applications,
Physical Data Independence - modify physical schema w/o changing the

### Database System Components
The Stored DB
The DBMS
The Applications
The Users

### Three-Schema Architecture


External level - provides access to particular parts of the DB to the users.
Conceptual level - structure of the DB.
Internal level - physical storage structures of the DB.



## MODULE 1

### Entity
Physical/conceptual object which has data associated with it. Has various attributes associated with it. Same entity can have different attributes depending on the system.

### Entity Sets
The collection of all entities of a particular entity type in the database at any point in time. Can be mapped to a table.

### Entity Type
Provides a format for the data which needs to be recorded to represent a particular entity. The entity type is described by its name and attributes.

### Key Attribute
A unique identifying attribute of an entity. It is distinct for every entity in the entity set.

### Several Attribute Keys
Composite key attributes can also be a unique identifier. Only the composition of all attributes in the composite attribute necessarily needs to be unique.

### Value Set of Attributes
specify the set of values that may be assigned to a particular attribute of an entity. A particular entity may not have an applicable value for an attribute (Null value can be used).

### Composite/Simple
Composite – an attribute composed of multiple other attributes (not atomic)
Simple – not composite – only one attribute which cannot be divided (atomic).

### Multivalued/Single Attribute
A multivalued attribute which has multiple values stored within it. A single attribute only has one value.

### Stored/Derived
Derived – when attribute values can be derived from related attribute values. Derived attributes are computed from stored values (e.g. age from DOB)

### Relationship Types
An association among two or more entities. A relationship type defines the relationship. May have descriptive attributes and key attributes.

### Relationship Degree
How many entities involved
Binary
Ternary
N-ary

### Entity Roles
Participating entities play a particular role in a relationship type. Role name specifies the role an entity plays.

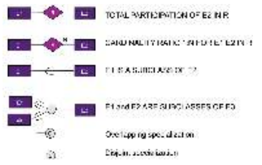
### Recursive Relationship
Self-referential relationship. An entity can participate more than once under different roles (e.g. manger/managed)

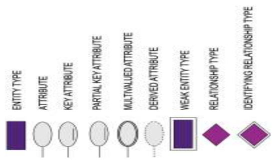
### Relationship Set
Collection of relationships of a particular relationship type at any point in time.

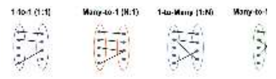
### Existence Dependency
Indicates whether the existence of an entity depends on its relationship to another.

### Weak entities
Entity types that do not have key attributes of their own. They must use their owner's PK and their partial key to form their composite primary key. Identifying relationships is the relationship a weak entity has with its owner.

### Specialisation/Generalisation
EER support subclasses.
Specialisation – subclasses (contains a subset of entities from the superclass).
Generalisation is the opposite – ignore distinguishing characteristics and generalise them into one superclass.

### Notation Guide

TOTAL PARTICIPATION OF E2 IN R
N
CARDINALITY RATIO 1:N FOR E1:E2 IN R
E1 IS A SUBCLASS OF E2
E1 and E2 ARE SUBCLASSES OF E3
Overlapping specialization
Disjoint specialization

### Notation Guide


### Cardinality Ratio


## MODULE 2

### Relations
Main structure for representing data – a set of records.

### Relation components
Relation name
Attribute Names
Tuples
Attribute Values from same Domain

### Domain Types
Domain is a set of atomic values (indivisible) each domain has a data type or format (e.g. int, char, datetime).

### Attributes
Each attribute A is the name of a role played by some domain D in the relation named R. The number of attributes in a relation is the degree of R.

### Domain/Attribute Restrictions
Same attribute name does not mean same domain.

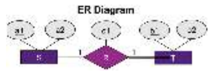
### Tuples
Ordered list of n values where each corresponds to the domain of attribute that it is a value for. Known as n-tuples.

### Relation Schema
Denoted by R
[A1,A2,A3,...AN]
Integer n is the degree of the relation

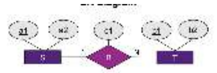
### Relation Instance
Denoted by r(R) is a set of n-tuples r = {t1,t2,tm}

### Entity
E[<u>a1</u>,a2,a3]

### Weak entities
E[<u>a1</u>,a2]
W[<u>a1</u>,<u>b2</u>,b3]
W.a1 references E.a1
T.a1 references S.a1

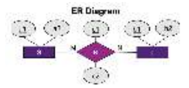
### Binary 1:1


S [<u>a1</u>,a2]
T [<u>b1</u>,b2,a1,c1
T.a1 references S.a1
One with total participation stores all simple attributes of the relation & FK to the other.

### Binary 1:N


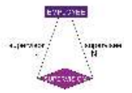
S [<u>a1</u>,a2]
T [<u>b1</u>,b2,a1,c2]
T.a1 references S.a1
Like before, any of the simple attributes must be stored in one of the relations (the N-side) & FK to the other.

## MODULE 2

### Binary M:N


S [<u>a1</u>,a2]
T [<u>b1</u>,b2]
R [<u>a1, b1, c1</u>, c2]
R.a1 references S.a1
R.b1 references T.b1
Create a relation for the relationship
Combination of the PKs of all participating entities becomes the PK.

### Sparse Relationship Mapping
Mapping 1:1 and 1:N as M:N only do this if you want less nulls as FKs.

### Recursive


Employee [<u>SSN</u>, fname, mlt, lname, dob, address, sex, salary, dnumber, superSSN]
Employe.superSSN references Employ.ssn
There should exist a super attribute of the key attribute.

### N-nary relationship
Same as M:N relationship

### Multivalued attributes
Like a weak entity

### Super & Subclasses
A relation is created for the superclass and the subclasses. The primary key of each of the subclasses in the primary key of the superclass. The relations of the subclasses include a foreign key from the primary key of the relation of the superclass entity type's