01

Date:⇒ [illegible]

# # Unit ⇒ 1 #

## Lecture ⇒ 1

Date:⇒ 03/08/19

## # Lecture ⇒ 1 #

Database Management System

(i). Data:⇒ Data are the facts or details from which information is derived.

→ Data are the raw fact 8 pieces of information with no content (meaning).

Ex ⇒ 50 → data – you would not have learned anything.

★ Data may be text, numbers, images, videos etc ★

(ii). Information:⇒ when data is processed, organized, structured or presented in a given context so as to make it useful, it is called information.

Ex ⇒: 50 no. of students registered for java class

[data with context)

## 3. Databases:⇒

DATABASE

- ↓ Data
- ↓ meanigful method for accessing & manipulating data

Database⇒: "A datab ase is a collection of related data from which users can effiectuely reterive the desired information".

Ex⇒: Diitonary → Key-value Pairs
word – meaning

method – access data using word (key).

~~non computerized database~~

Database →
- → Collection of name, roll no, address. of student
- → collen of Sound
- → collen of images
- → Collen of videos.

DATABASE

Computarized
- → Dictionary
- → Recipe Book
- → Tv Guide

Non-computarized
- → Customer files
- → employee
- → information

→ Database are organised, they have a structure, & all the data they store fits into that structure.

→ Database are quite similar to spreadsheets, as they are mostly made up of tables which contains rows & column.

Student database

| Roll NO | Name | Address | Sem | Branch |
|---|---|---|---|---|
| | | | | |

## 4. Database Management System:⇒

A DBMS is a integrated set of programs used to create & maintain a database.

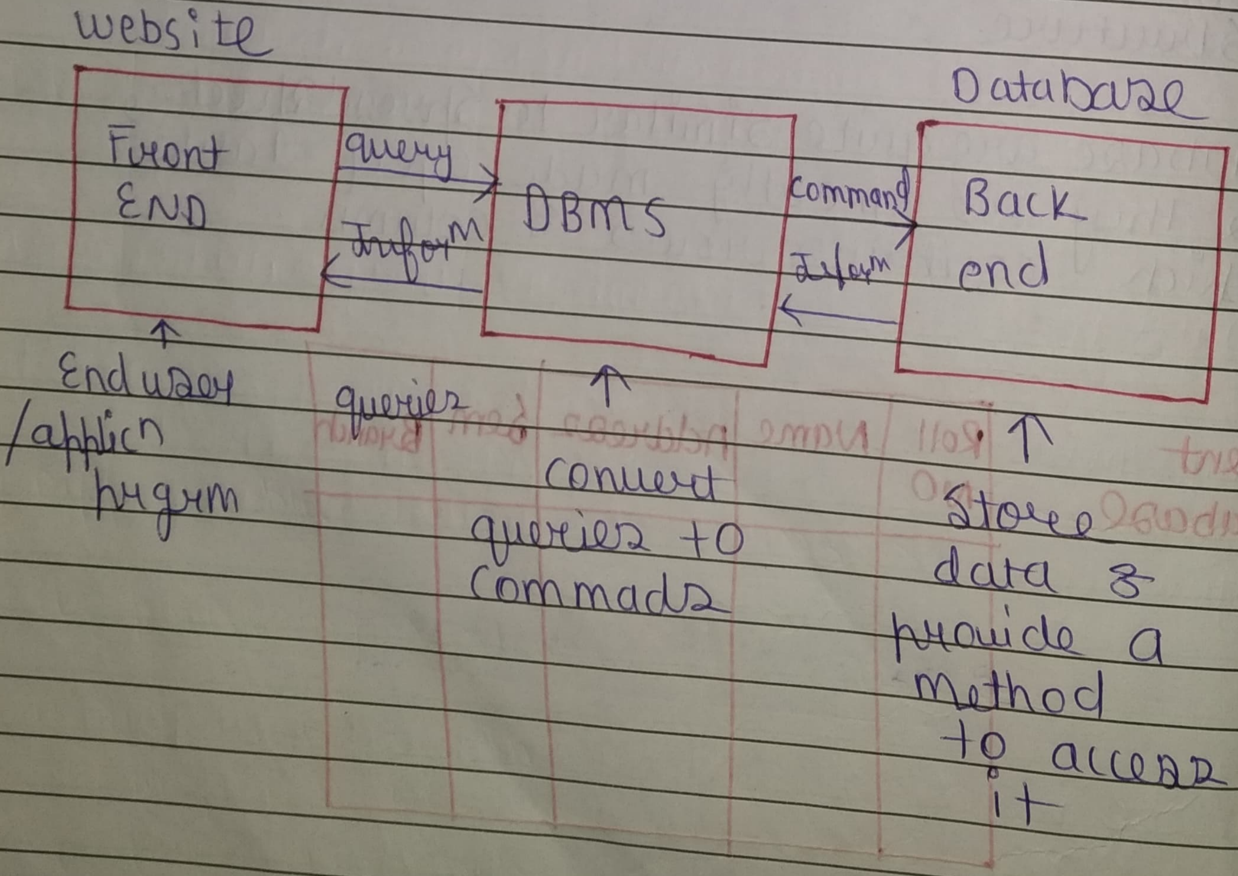
main objective of DBMS is to provide convienent & effective method of

→ Defining → Manipulating

→ Reterving

→ Storing

the data contained in the database.



Basic information advantages & disadvantages read from website [or other sources].

# Requirement from Data → : consider Ex → for fb

→ to meet several requ. from end users

| Requirement | Description |
|---|---|
| → Integrity | Data should be accurate<br>Ex → my fb profile should contain Valid Country name. |
| → Availability | I should be able to access fb & see my data at all times |
| → Security | Only my friends should be able to see my Posts & no one else |
| → Independent of Application | → same Access from Android app as well as from web browser on lappy |

| | |
|---|---|
| → Concurrent | All my friends should be able to see my posts at same time |

# Flat file→: A flat file datbase that stores ⇒ data in a plain text file.

→ Each line of text file holds one records, with field seperated by delimiters, such as commas or tabs.

Ex→: most database program such as MS-Access & filemaker Pro can import flat file databases & use them in larger relational databases

# limitations of flat file→:

✓ ★★ Read diffⁿ from previous copy Notes.

→ Data is stored in flat files & can be accessed using any programming language.

→ The file base Approach suffers following problems:→

1 → Dependency of program on physical structure of data.

2 → Complex process to reterive data

3 → loss of data on concurrent access

4 → Data redundancy.

5 → Security.

# Function of DBMS:→

** Read from previous notes & mam videos.

# Types of database System:→

→ Categorized into 4 types→:

→ based upon the underlying structure used to store data.

→ They are in chronological order of evolution

are → Hierachical
Network
Relational
Nosql.

# Relational Model:→

→ Relational database store data in relation i.e. tables.

Relation is usually represents as.

Employee (ID, ENAME, SALARY, BONUS, DEPT).

Attributes / columns / fields

| ID | ENAME | SALARY |
|---|---|---|
| 1 | lokesh | Infinite (1400,0000) |
| 2. | lokesh! | 200000 |

Schema Create see this topics from [illegible] → Data models abstraction. 05. [illegible]

Date:→ 01/06/20 # Next Topic #

# Schema & instance →:

Schema →:

→ Description of database is called database schema.

→ The overall design of database is called schema & database schema is specified during database design & not expected to change frequently.

There are three types of schema →

(i). Physical schema / Internal schema

(ii). Logical schema / Conceptual schema

3. View schema / External schema

→ A D.S. is designed by the database designer to help programmer whose slw with interact with database.

Schema represents the logical view of entire database

Ex→:

Student

| Name | Roll no | Branch | Sem |
|---|---|---|---|

Course

| Course no | C Name | Department |
|---|---|---|

# Database instance →

→ The data stored in database at a Particular moment of time is called instance of database.

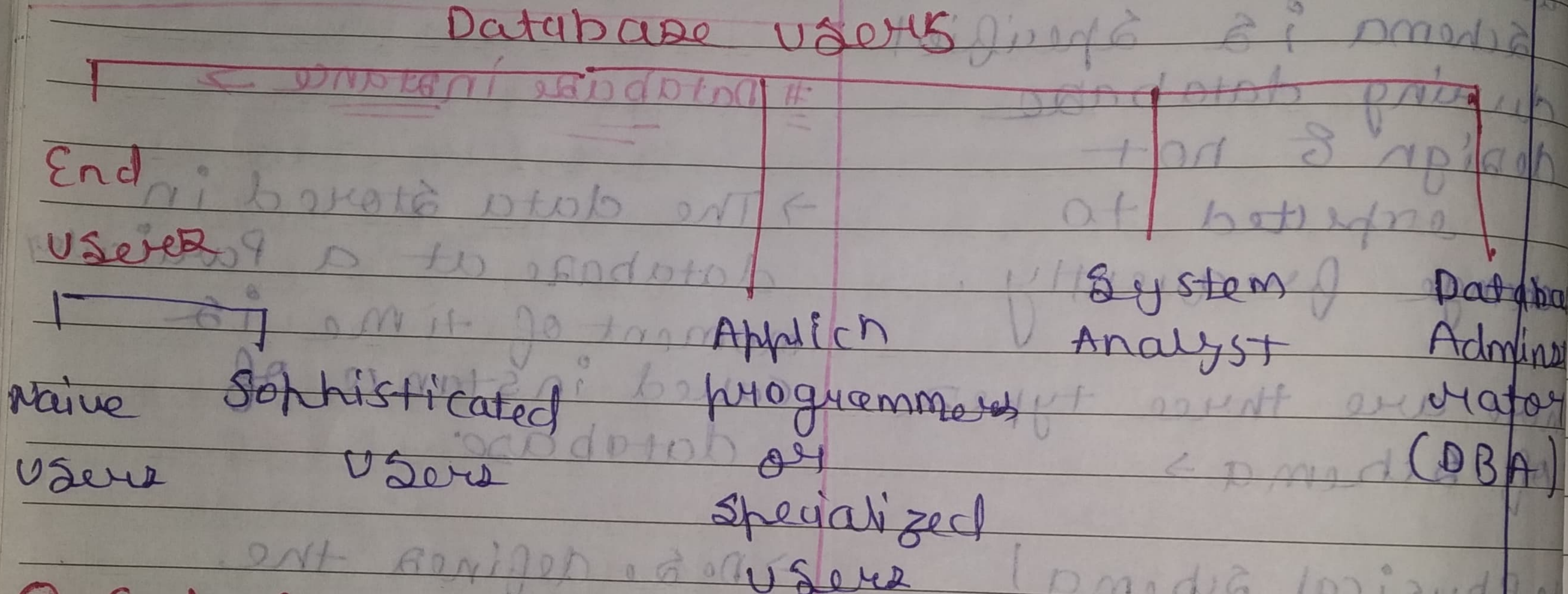
→ D.S. defines the variables in tables that belongs to Particular database.

→ The values of these variables at a moment of time is called instance of database

→ Instance of database changes over time when we add or delete from database.

# Types of Database Users →:

→ There are no. of database users who interact with the database in order to access & update the database.


Database Users
End Users
Naive Users
Sophisticated Users
Applicn programmers or specialized users
System Analyst
Database Administrator (DBA)

①. End Users → are those who access the database from the front end. They use the developed applications & they donot have any knowledge about the design & working of database.

→ Their main aim is just to get their task done.

①. (i). N.U → is any user who doesnot have any knowledge about database. Their task is to use the developed app & get desired

Ex→ of Naive User→ owner of the bookstore who enter the details of various books in the database by using an appropriate application program

### iii. Sophisticated Users→

→ S. Users have great knowledge of query language so they use database query language to access information from the database to meet their complicated requirement

Ex→ Users such as Business Analyst & Scientist interact with the database

### ②. Application Programmers→:

→ Specialized users write application program that uses the database.

→ Application programs can be written in any high level pro. lang. like C#, JAVA, .net

→ Specialized users interact with the DBMS through DML.

### ③. System Analyst→:

S.A. is responsible for the design, structure & propertie of database.

→ Specification given by the S.A are implemented by A.P. to develop application program which are used by end users.

## 1. Database Administrator (DBA):->

-> DBA can be a single person or a group of person

-> DBA has central Over both data & Application programs

-> DBA is responsible for every thing that is related to database. He makes policies & strategies & provide technical support.

Role / Responsibilites of DBA ->:

(i). Schema definition & modification

2. Granting of Authorization for data access

3. Database Availability

4. Training & Supporting users

5. Database Availability

## # Database languages & interfaces

DB languages are used to create & maintain a database on computer

Ex -> Oracle & Sql, dBase, MS Access.

D.L. (SQL statements)

DDL ↓ Definition

DCL ↓ control

DML ↓ Manipulation

* * *

Description about all

Data independence→ The ability to modify a scheme definition in one level without affecting a schema defⁿ in the next higher level is called D.I.

2 types → Physical independence
→ logical independence

# ER Data model→:

→ It is the most popular conceptual model or object based model used for designing a database.

→ ER data model views the real world as a set of basic objects (entities), their attributes & relationship among these objects (entities)

✓ ** See Data model video

→ Entities, attributes & relationships are the basic construct of ER-model

Date→ 02/06/20

→ER d.m. describes the structure of a database with the help of a diagram called ER Diagram.

Entity →: An entity is an object or distinguishable thing in real world.

Ex → car & student, People etc.

Attribute:→ Each entity has certain characteristics.

Ex → Student has attribute Name, age, marks etc.

# Relationships→:

Specify relations among entites from 2 or more entity sets.

Ex → Relationship

↑

Company has Employees

↓ ↓

entity entity

Entity → Table (relation)
Attributes → Columns

## Entity type →:

→ Collection of entites that share the same attributes but different values is known as E.T.

Entity Type

↓ Strong entity type

↓ weak entity type

(ii). Strong entity type:→

An entity type that has a key attribute which uniquely identifies each entity is called a strong entity type.

→ Also known as 'independent entity'.

Ex→ STUDENT is a strong entity type which has Rollno key attribute.

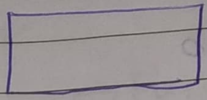
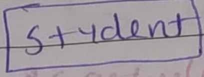
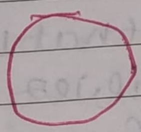
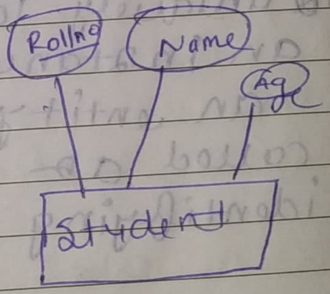
(2). weak entity type:→

→ An entity type that doesn't have any key attribute of it's own (~~[illegible]~~ dependent entity).

Ex→ Edition is a weak entity it depends on another entity type Book for it's existence

Agar Book hee nhi hogi toq Edition bhi nhi hoga.

ER diagram notations / symbols

| Symbol | Purpose |
|---|---|
| (i). Rectangle<br><br>Ex→<br> | Represent entity type |
| 2. Oval / ellipses<br> | Represent attributes<br>Ex→<br><br>Rollno→ key attribute<br>→ Uniquely identifies ~~[illegible]~~<br>Key attr. (—) |

# Attributes →

→ are the properties of an entity to describe it

Ex →:

Student (entity) → RollNO<br>→ Age<br>→ class<br>(Attributes)

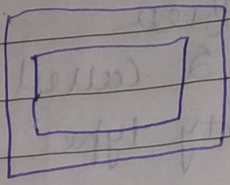
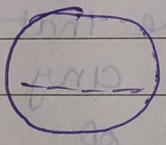
# Types of Attributes →:

(i). Identifying Attributes →:

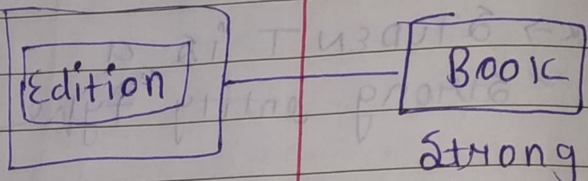
→ The attribute that used is uniquely identifies an instance of an entity is called as identifying attribute.

Ex →:

RollNO of the entity type STUDENT is an identifying attribute as no two student can have same rollno.

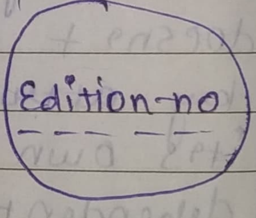
| Symbol | Purpose |
|---|---|
| → Double Rectangle<br> | Represent weak entity type |
| →  | Represent Partial key of weak entity type |

Ex →



weak entity

strong entity.



## (iii). Descriptive attributes → (descriptor)>.

→ The attribute that describes a non-unique characteristics of an entity instance is called as descriptor'

Ex→: Name & age are the descriptive attribute of STUDENT entity type as two students can have the same name & age.

## 3. Simple Attribute:→

→ The attributes which cannot be Partitioned into smaller Sub Parts is called Simple attributes.

Ex→ Book_title, price, Year attributes of Book entity type are simple Attributes.

## 4. Composite attribute:→

→ A C.A. can be subdivided into smaller subParts which further form attributes

Ex→:

Name
- → First Name
- → middle Name
- → last name:

## 5. Single valued attributes:

→ The attributes that can have only one value for given entity are called Single valued attributes

Ex→ Book title is a Single V. A bcoz as one book can have only one title

## 6. Multi-valued Attributes →:

→ The attribute that can have multiple value for given entity are called multi-valued attributes.

Ex→ STUDENT are multi-valued attributes bcoz it has phone & email

## 7. Stored & derived attribute →:

A derived attribute calculate it's value from another attribute

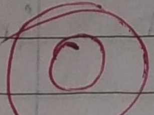
Ex→ The value of attrib Age can be determined from the current date & value of DOB attribute

Age→ derived Att'r.

DOB→ Stored Att.

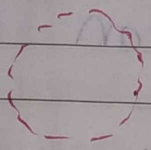
Symbol

Double oval



Represent multivalued attribute

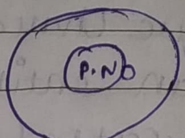
Dashed ellipse

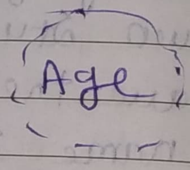


Represent derived attribute

Purpose

Ex→





# Relationship→:

→ Association or attachement between two or more ~~[illegible]~~ entites.

→ It describes how two or more entites are related to each other.

Ex→

Teacher teaches a Student.

# Degree of Relationship→:

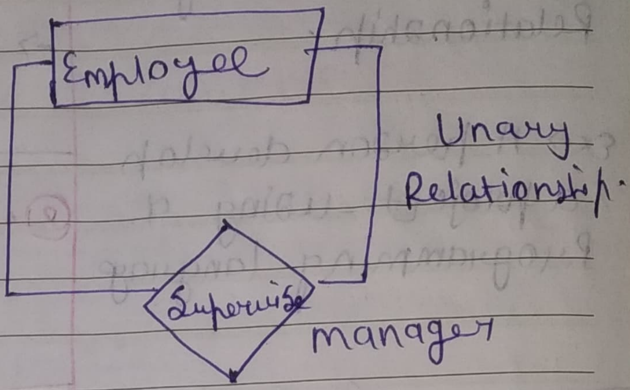
→ It signifies the no. of entites involved in a relationship.

(i). Unary Relationship→

If only single entity is involved in a relationship then it's a unary relationship.

Ex→

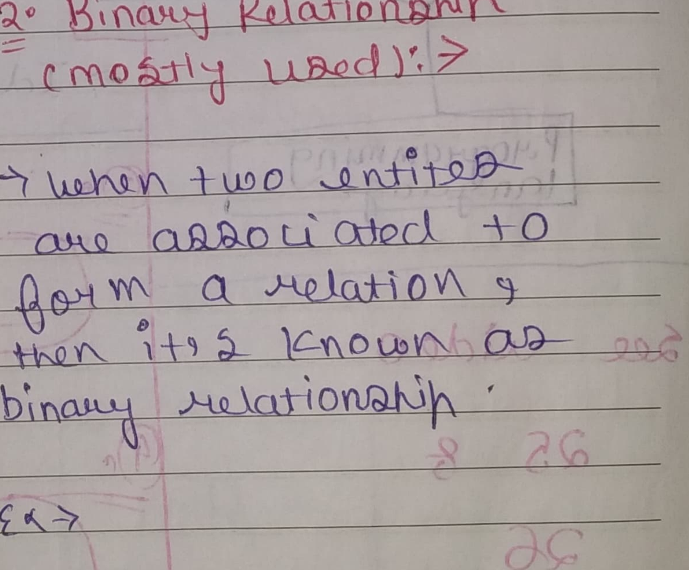
An employee (manager) Superuise another employee



2. Binary Relationship (mostly used):→

→ when two entites are associated to form a relation & then it's known as binary relationship.

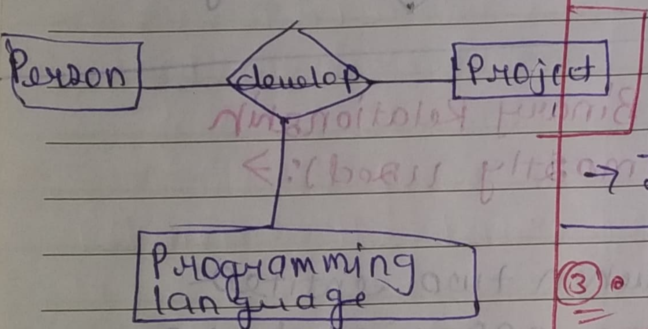
Ex→



3. Ternary Relationship:-

→ when three entites are associated to form a relation then its known as ternary Relationship.

Ex→ A person develop a project using a Programming language



See video

25 &

26

✓

*** Remaining

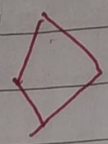
&

mapping

cardinality

# ER diagram Notation

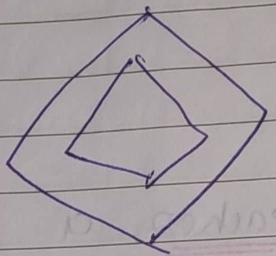
①.



(Diamond)

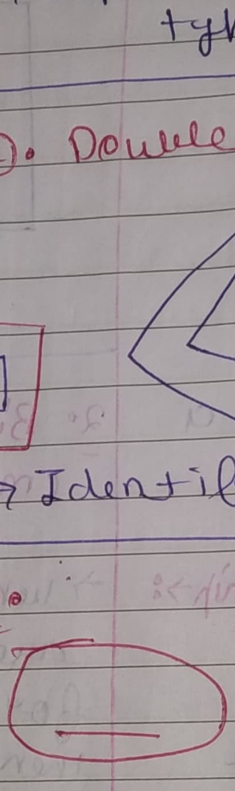
→ Represent relationship type

②. Double Diamond→



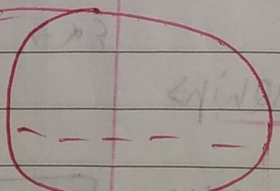
→ Identifying relationship.

③.



→ Represent key Attribute

④.



→ Represent partial key of weak entity type

See defn of Generalization, Aggregation & Specialization

# # Relational data Models #

→ In relational data model the data & relationships are represented by collection of interrelated tables (or relations).

Table consists of row & columns.

Column ~~represent~~ attribute of an entity.

Rows represents records.
/tuples

Ex→ STUDENT relation

→ Attributes

| Rollno | Name | Age | Address |
|---|---|---|---|
| 1 | L | 21 | IND |
| 2 | L | 21 | IND |
| 3 | B | 20 | IND |

→ Rows / tuples

→ key attribute.

Schema / Description of Relation → A relation schema represents the name of relation with its attributes.

Ex→ STUDENT (RollNO, Name, Age, SGPA }

→Tuple→ A Row of a relation is called as tuple which is ordered set of valuees.

→ Tuple contains a single record.

→ key→ Each reln contain attribute or Set of attributes that uniquely identifies each row (record) of a relation (table) is called as a key.

→Relation→ The table is called Relation

→ A Relation is a set of tuples (rows) & attributes (columns).

→ Domain→

A domain is a set of allowable value for one more attribute.

Ex→ Domain of Roll no is set of character String of length 12

| 0875CS171075 |.

Record / tuple / row are same 20. | Table / Relation | Field / attribute / column

# Degree of a Relation

→ Total no. of attributes in a Relation.

→ Cardinality of a Relation →:

→ Total no. of row present in a relation.

Date:→ 03/06/20

# Key →: A minimal set of attributes which can uniquely identified the tuples in a relation are said to be key attributer or simply key.

→ Key should be unique & not null.

# Types of key's →:

① • Super key
② • Candidate key
③ • Primary key
④ • Alternate key
⑤ • Composite key
⑥ • Foreign key.

(i) Super key:→

→ A Super key is any combination of attributes within a relation that uniquely identify each record / tuple within that relation.

Ex→

STUDENT (Rollno, Name, Address)

Superkey → Rollno
→ Rollno, Name
→ Rollno, Address
→ Name, Address

★ more than one Superkey in relation is Possible.

## 2. Candidate key :-

- A C. key is an attribute or set of attributes in a relation that are eligible to become a primary key.

- A reln can have more than one candidate key

- A C.k. is unique & not null.

- C. keys are minimal super keys.

Ex :-

S. key

Roll No<br>Roll No, Name<br>Rollno, Address<br>Name, Address →

Candiate key

1. Roll no
2. Name, Address

- Each C. key can work as primary key.

## 3. Primary key :-

A primary key is a set of attribute or one or more attributes of a relation that uniquely identify a record / tuple of a relation.

- Primary key attribute has no duplicate & null values.

- Only one candidate key can be a primary key

Ex

| Candiate key | Primary key |
|---|---|
| 1. Roll no | 1. Roll no (Rarely changed attribute) |
| 2. Name, Addr | |

## 4. Alternate key →:

→ Candidate key that are not selected to be the primary key are called as Alternate keys.

Ex→ Candidate key.

[ Rollno

Name, Address

Alternate key → Name, Address

## 5. Composite key →:

→ when a single attribute can't be used for unique identification, then a combination of attributes are used as a key such a key is called Co key.

Ex→

STUDENT (Rollno, Fname, Lname, DOB)

Composite key

Rollno
Rollno & Fname
Rollno, Lname
Rollno, DOB
Roll, Fname, Lname
Fname, Lname, DOB

↓

composite key

→ A P. key that made up of more than one attribute is known as Co key.

Ex 2→

WORK (PID, EID & Hours_worked)

| EID | PID | Hours-worked |
|---|---|---|
| 1 | 1 | 600 |
| 1 | 2 | 800 |

Co key (EID, PID)

## 6. Foreign key →

→ Foreign key is the attribute of relation that is primary key in another relation.

→ Fo key is used to establish relationship b/w two relations (tables).

→ F. key can have null values & duplicate value as well

Ex →

BOOK

| ISBN | Title | Price | Pages | P-ID |
|---|---|---|---|---|
| 234-11 | C | 200 | 300 | P001 |
| 341-22 | C++ | 250 | 400 | P001 |
| 333-111 | Python | 500 | 600 | P002 |

↑ Primary key

P-ID → Foreign key

Publisher

| PID | Pname | Addr |
|---|---|---|
| P001 | XYZ | INDB |
| P002 | ABC | RTM |

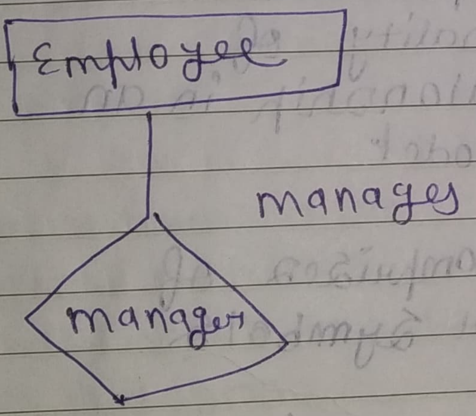
↑ P. key

24.

# Self-Referening F. key or Recursive foreign key →:

→ A f. key that refers to its own relation is called self-referening f. key

→ Dusri table ki f. key ko use na karke ape table ki foreign key use kare vo Self-Referening F. key !



→ To show unary relationship we use Recursive F. key

| ESD | manager |
|---|---|
| E01 | E10 |
| E02 | E07 |
| E03 | E07 |
| E04 | NULL |

# See Relational database video 41

& See ACID properties

# Data integrity & constraints

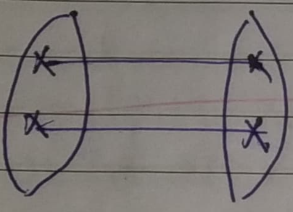
✓ See this topic

# Cardinality of Relationship →:

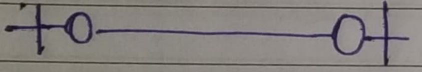
→ is the no. of instances in one entity which is assocciatd to the no. of instances in another.

→ Three are 3 types 1:1, 1:N & M:N

(i).

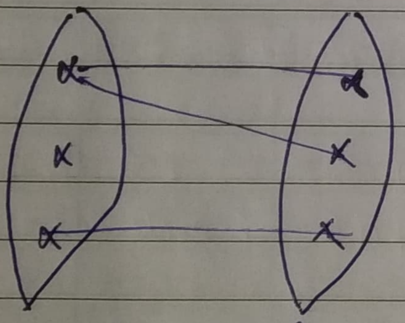


(1:1)

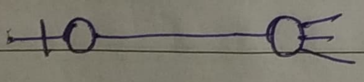


(1:1) One to one

(ii).

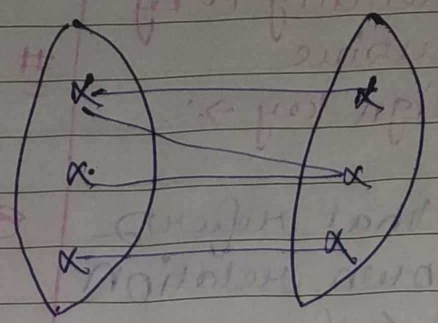


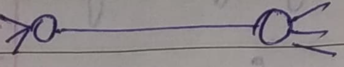
1:N (one to many)



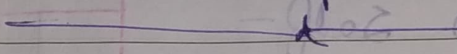
M:N

Many to many





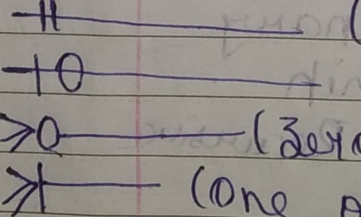
Ex → Relationship b/w Employee & computer



# Crow - foot Notation →:

→ is one of the ways to represent cardinality of relationship in an ER model.

→ It comprises of four symbols



(one (Exactly one))

(zero or one)

(zero, one or more)

(One or more)

# SQL :→

→ is a standard language for accessing & manipulating databases.

→ SQL stands for structured Query language

→ SQL becomes standard of the ANSI, in 1986 & ISO in 1987.

→ what can SQL do →:

SQL can execute queries against a database

can reterive data from database.

can insert records in database

update, delete, create new database.

new tables create views in database

set permission on table, procedure & views

→ Using SQL in your web site

→ To build a web site that shows data from database, you will need-

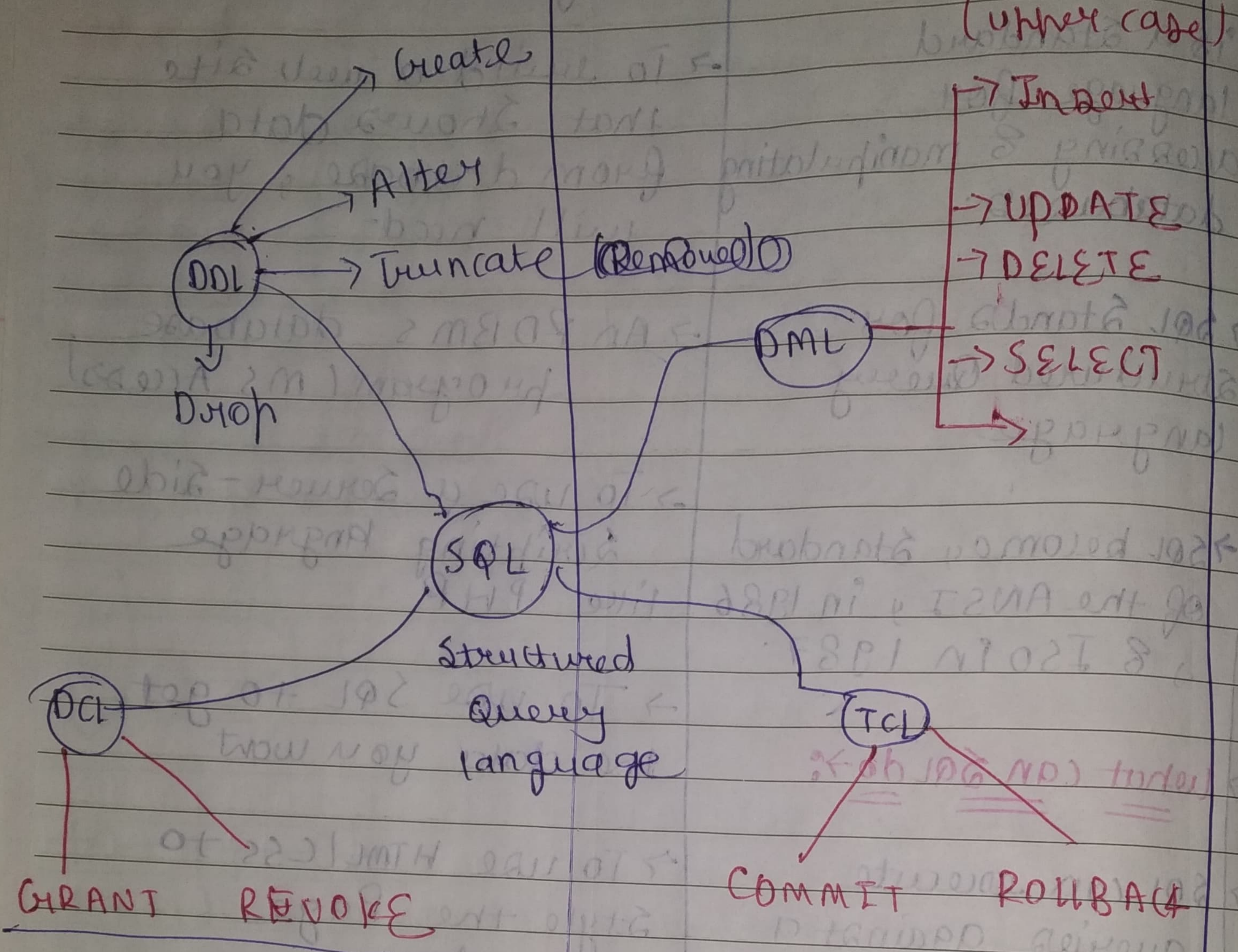
→ An RDBMS database program (MS Access)

→ To use a server-side scripting langrage like PHP

→ To use SQL to get the data you want

→ To use HTML/CSS to style the Page

ASP



(i). DDL →: Data definition L

→ Create →: Create new database objects

→ ALTER →: Modify existing database objects.

→ DROP →: Delete existing database objects

→ TRUNCATE →: Remove all Rows from Table

(ii). DCL → Data control l.

→ GRANT → Provide access rights on database

→ Revoke → withdraw access rights on databases.

③ • DML: →

3/7 5/6
45 0

28.

65 7 5
(60+5) 7 5 015
5x75
100

3. DML:-> Data Manipulation language

INSERT -> create new rows in tables

UPDATE -> modify data in tables

DELETE -> delete data from tables

SELET -> Reterive data from tables

4. TCL->: Tranaction Control language

-> commit -> save database changes & end tranaction

-> Rollback -> undo changes that are committed & end tranaction

Binary large object

→ Storing large binary data like movies, images with size up to 4GB.

Date→06/06/20

Sql Data Types

→ Numeric → decimal, numeric, float, real, int, bit, tinyint, bigint

→ Date/Time → Date, Time, Date Time, Timestamp, Year

→ Character / String → char, Varchar(max), Text.

→ Unicode Character / String → Nchar, Nvarchar, N text, Nvarchar(max)

→ Binary → Binary, Varbinary, image, varbinary(max)

→ Miscellanous → clob, blob, xML, JSON

** See description on Journal dev website

# Arithmetic operators :

Same as Python
but one difference in

| Operation | Python op. | Sql op. |
|---|---|---|
| Assignment | = | = |
| Equality check | == | = |

L.L → Lower limit

U.L → Upper limit

# Other comparison operator :

| Operator | Symbol | Usage | Example |
|---|---|---|---|
| Range | BETWEEN <L.L> AND <U.L> | matches values b/w a Range | |
| List | IN (list of values) | matches any of a list of values | DEPt IN ('IVS', 'ICP') |
| String Pattern matching | LIKE | matches a character pattern | Supplier Id LIKE 'S%' |
| NULL Test | IS NULL | Is a null value | Bonus IS NULL |

Logical operators

| Operator | Symbol | Usage | Example |
|---|---|---|---|
| AND | AND | Return True if both are True | |
| or | OR | Return True if any one is true | |
| Not | NOT | Return T if following condn is False. | |

# Create and Drop Table ->:

-> CREATE TABLE Statement is used to create a table in a database.

-> Database tables are organized into rows & columns

-> Each table must have a name & can have any no. of columns (minimum 1 column) is required.

-> Each column must have a data type that determines the type of value that can be stored

-> All tables must have a unique name

-> DROP TABLE Statement is used to remove an existing table from database.

Ex ->:

Table name

Data type

```
CREATE TABLE Student (
    StudentId  INTEGER,
    FName      VARCHAR2(10),
    DOJ        DATE,
    Gender     Char(10)
)
```

size

-> Column Name

Ex ->:

```
DROP TABLE Student
```

Note:>

Column names should be seperated by commas

-> No two columns can have the same name.

Previous Table O/P →

| STUDENT ID | FNAME | GENDER | DATE |
|---|---|---|---|
| 1001 | 'Alex' | 'M' | '23Jan-2015' |

→ when we run this

→ Run

DROP TABL Student; → Ex of Table Drop

# Constraints:→

→ various constraints that can be created on database tables are:→

- → NOT NULL
- → PRIMARY KEY
- → CHECK
- → UNIQUE
- → FOREIGN KEY.

→ SQL Constraints:→ SQL Constraints are used to specify rules for data in a table.

→ Constraints are used to limit the type of data that go into a table.

→ Constraint are classified into multiple types based on the no. of Columns they act upon as well as they Specified

| Constraint Type | Applies on |
|---|---|
| Single Column constraint | Single column |
| Composite Constraint | multiple column. |

| Constraint Type | Specified | |
|---|---|---|
| Column level constraint | with column $def^n$ | → apply on column |
| Table level constraint | After Column $def^n$. | → apply on table |

Date:-> 07/06/20

# Not Null Constraints:->

By default column can ~~have~~ hold Null value

-> Not Null constraints prevent a column from accepting NULL values.

-> Not Null can be applied as a column level constraint.

** constraint name is optional & it can be specified by using constraint keyword.

Ex->
Let us now create NOT NULL constraint on StudentId and FName columns.

```
CREATE TABLE Student (

StudentId INTEGER CONSTRAIN Stud_SId_nn NOT NULL,
FName VARCHAR2(10) NOT NULL,
LName VARCHAR2(10) );
```

O/P->

| Name | NULL? | TYPE |
|---|---|---|
| STUDENTID | NOT NULL | NUMBER |
| FNAME | NOT NULL | VARCHAR2(10) |
| LNAME | | VARCHAR2(10). |

# Default:> A column can be given the default value by using DEFAULT option.

→ The data type of column & default expression must be the same.

→ DEFAULT option can be provided for nullable as well as NOT NULL attributes.

Ex >

```
CREATE TABLE Student (
  StudentId  INTEGER,
  FName  VARCHAR2(10),
  DOJ  DATE  DEFAULT  SYSDATE);
```

O/P >

| Name | NULL? | TYPE |
|---|---|---|
| StudentId | | NUMBER |
| FName | | VARCHA2(10) |
| DOJ | | DATE |

⇓

Insert values

| STUDENTID | FNAME | DOJ |
|---|---|---|
| 1001 | Lokesh | 07-JUN-2020 |

# Primary Key Constraint:>

→ Pokey constraint on a column ensures that the column can't contain NULL & duplicate values.

→ we have only one Primary key in a table.

Ex >

we will now create a Student table with primary key constraint on StudentId column

⇒ ⇒ ⇒

Ex >

CREATE

StudentId

Stud

— ''

— ''

# Check

→ CHECK

used

value

speci

col

Ex >

Cond

CONSTR

Ex>

```
CREATE TABLE STUDENT (
StudentId INTEGER CONSTRAINT
  Stud_Sid_PK PRIMARY
        KEY,
  - "  ——— "
  - "  ——— );
```

# Check Constraint :>

-> CHECK Constraint is used to limit the values that can be specified for a column.

Ex>

```
Gender CHAR(1)

CONSTRAINT Stud_gender_CK
   (Gender IN ('M', 'F'));
```

# Unique Constraint :>

-> A unique constraint on a column ensures that two rows in a table can't have same value in that column.

-> Unique constraint allow NULL values.

-> A table can have many UNIQUE constraints

Ex> Create UNIQUE constraint on ContactNO so that the two Student can't have the same contact details.

Ex>

```
        (

ContactNO    NUMBER(10)
CONSTRAINT  Stud_CNO_UK
   UNIQUE);
```

# Need for Foreign key ->:

Student Table has already been created & inserted with few records in the database.

Ex:-> Student Table.

| STUDENTID | FNAME | Contact NO |
|---|---|---|
| 1001 | Alex | 922964733 |
| 1002 | John | 625114092 |

when we insert

course ID & marks scored in new table

so acc. to Student Id

| | | |
|---|---|---|
| 801 | 1001 | 78.5 |
| 802 | 1002 | 90.75 |
| 801 | 2001 | 85 |

-> The student with id 2001 is not a valid value as it's not present in the Student Table.

-> To avoid this problem & the relationship has to built b/w Student & marks table.

-> This can be acheived using Foreign key.

# Foreign key constraint:->

```
(
StudentId INTEGER
CONSTRAINT marks_sid_
FK
REFERENCES Student
(StudentId)
```

queries see from w3 school

# Day → 4 #

Reman 021
DML statements

Date:⇒ 11/06/20

## # SQL functions:→:

→ SQL functions are built - in modules provided by database

→ You can use them in DML statements to perform calculations on data.

→ All functions returns a single value.

→ They are categorized into two types based on a no. of rows they operate upon

|  | Single Row Function | Multi-Row function |
|---|---|---|
| Returns | Single Row | Single Row |
| Operates on | S.R. | Multiple Row |
| used in clauses | Select, where, order by & having | Select & ORDER BY AND HAVING Clause |

→ Single row function produces one row of o/p for each row of I/P.

| ENAme | Salary |
|---|---|
| James Potter | 75000 |
| Ethan C. | 90000 |

upper (EName)

| EName |
|---|
| James Potter |

| EName |
|---|
| Ethan M. |

multi-row function produces just one row of o/p, irrespective of the no. of I/P

| Ename | Salary |
|---|---|
| James P | 75000 |
| Ethan m | 90000 |

Sum (Salary)

| Salary |
|---|
| 165000 |

# Numeric functions:>

→ N.F. are single row functions that accept a numeric value & return numeric o/p.

①. ABS → ABS(value)

→ Return abs value of a no.

②. Round → Round(value, digit).

③. CEIL → CEIL(value)
→ Round up the fractional value to next integer.

④. Floor(value) → Rounds down the fractional value to the lower integer

# Character functions:>

→ character functions work on character strings & can return a character string or numeric value.

①. UPPER
②. LOWER
③. CONCAT
④. LENGTH.

→ Substring function is used to extract part of a string.

Syntax →

SUBSTR(value, start_position, length)

Ex:

Substr Ex →

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| D | A | T | A | B | A | S | E |

SUBSTR('DATABASE', 5)

O/P → BASE.

# Conversion functions:->

-> use conversion function to convert data from one format to another.

1. TO_CHAR

-> TO_CHAR(value, format)

-> Converts a no. or date to a string.

-> use this function for formatting dates & numbers.

2. TO_DATE -> converts a string to date

3. TO_NUMBER ->

-> converts a string to a number.

# Data functions ->

-> is used to determine the current time & to perform date operations like adding a specific duration to a date & finding time diffn b/w two dates etc.

1. SYSDATE ->

Returns current date of system i.e. the host on which database server is installed.

2. SYSTIMESTAMP ->

Returns current timestamp of the system.

3. ADD_months (Add n moths to the given date).

4. MONTHS_BETWEEN

-> finds difference b/w two dates in months.

# Aggregate functions →:

→ A.f. operate on multiple rows to return a single row.

→ Some A.F like

SUM(total)
AVG(Average)
operates on numeric column while others like

MIN(lowest value),
MAX(highest value)
&
COUNT(no. of rows)
operates on all data types

→ All A.f. ignore NULL values except Count(*).

# miscellanous functions

①. NVL →

NVL(value1, value2)

→ subsitute value 1 by value 2 if value1 is NULL.

→ The data of value1 & value2 are same (must be same)

②. USER →
Returns the current logged in user.

# order by clause ✓ ** Remaining
→ Group by
→ having
✓ ** Union
023 Rem
Also (Day-48.5)

Date:→ 12/06/20

# Day → 5

# Join introduction: →

→ How do we fetch data from multiple tables in a single query.

→ Let us say we want to display employee id, employee name along with computer id, model of the computer allocated to the employee in a single tabular format.

→ We can meet such requirements using JOINS which can combine data from multiple tables

→ Joins are of multiple type: →

⇒ ⇒ ⇒ ⇒ ⇒
on next Page

For Ex →:

Employee Table

| ID | ENAME | COMPID |
|---|---|---|
| 1 | James | 1001 |
| 2 | Potter | NULL |

Computer Table

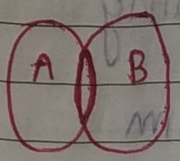
| COMPID | MODEL |
|---|---|
| 1001 | VOSTRO |
| 1002 | Precision |

Result Table

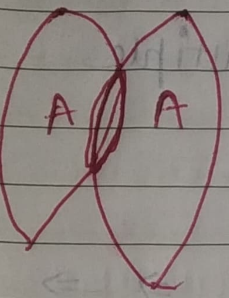
| ID | ENAME | COMPID | Model |
|---|---|---|---|
| 1. | James | 1001 | VOSTRO |

at the of Revisi

# Types of Join:→

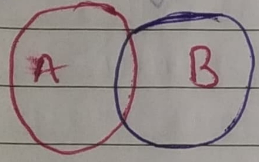


→ Inner Join

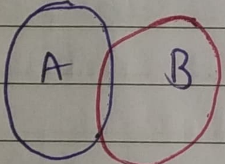


→ Self Join

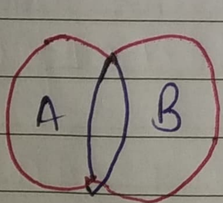
← Red for left



→ Left outer Join



→ Right outer Join



→ Full outer Join

[→ We have Cross Join → called Cartesian product]. → which is rarely used.

## ①. Cross Join:→

→ Also referred to as Cartesian Product.

→ A cross join with m rows in table A & n rows in table B will always produce m*n rows.

→ Essentially it combines each row from the first table with each row of the second table.

## ②. Inner Join→:

→ most frequently used Join.

→ It matches the record from both tables based on the join predicate & returns only the matched rows.

** → First a Cartesian product is created & then all the rows that don't meet the join condition are dropped from the result

✓ ** See all Join examples at time of revision

990

from infyta 8 w3 school.

# Self-Join:>

→ It represents join of a table with itself.

# Left-Outer Join>:

→ L.O.J for tables A and B will always return all records from table A even if matching record is not found in table B as per the join condition.

# Right-outer Join:>

→ Just opposite of left - Outer Join.

# Full-outer Join:>

→ Combines the effect of both left O.J. 8 R.O.J. Full.O.J b/w table A 8 table B returns matched as well as unmatched rows from both tables

# order of query Execution>

F J W G H S D

O

→ From
→ Join
→ where
→ Group BY
→ HAVING
→ SELECT
→ DISTINCT
→ ORDER BY