



Department of computer science and engineering

Regulation 2021

II Year- IV Semester

CS3492 - Database Management System Syallabus



PIE Tech

POLLACHI INSTITUTE OF ENGINEERING AND TECHNOLOGY
(Approved by **AICTE** and Affiliated to **Anna University**) *sky is the limit*

CS3492 DATABASE MANAGEMENT SYSTEM L T P C 3 0 0 3

OBJECTIVES

- To learn the fundamentals of data models, relational algebra and SQL
- To represent a database system using ER diagrams and to learn normalization techniques
- To understand the fundamental concepts of transaction, concurrency and recovery processing
- To understand the internal storage structures using different file and indexing techniques which will help in physical DB design
- To have an introductory knowledge about the Distributed databases, NOSQL and database security

UNIT 1 RELATIONAL DATABASE 10

Purpose of Database System – Views of data – Data Models – Database System Architecture – Introduction to relational databases – Relational Model – Keys – Relational Algebra – SQL fundamentals Advanced SQL features – Embedded SQL– Dynamic SQL

UNIT 2 DATABASE DESIGN 8

Entity-Relationship model – E-R Diagrams – Enhanced-ER Model – ER-to-Relational Mapping – Functional Dependencies – Non-loss Decomposition – First, Second, Third Normal Forms, Dependency Preservation – Boyce/Codd Normal Form – Multi-valued Dependencies and Fourth Normal Form – Join Dependencies and Fifth Normal Form

UNIT 3 TRANSACTIONS

Transaction Concepts – ACID Properties – Schedules – Serializability – Transaction support in SQL – Need for Concurrency – Concurrency control – Two Phase Locking- Timestamp – Multiversion – Validation and Snapshot isolation– Multiple Granularity locking – Deadlock Handling – Recovery Concepts – Recovery based on deferred and immediate update – Shadow paging – ARIES Algorithm

UNIT 4 IMPLEMENTATION TECHNIQUES 9

RAID – File Organization – Organization of Records in Files – Data dictionary Storage – Column Oriented Storage– Indexing and Hashing – Ordered Indices – B+ tree Index Files – B tree Index Files – Static Hashing – Dynamic Hashing – Query Processing Overview – Algorithms for Selection, Sorting and join operations – Query optimization using Heuristics - Cost Estimation.

UNIT 5 ADVANCED TOPICS 9

Distributed Databases: Architecture, Data Storage, Transaction Processing, Query processing and optimization – NOSQL Databases: Introduction – CAP Theorem – Document Based systems – Key value Stores – Column Based Systems – Graph Databases. Database Security: Security issues – Access control based on privileges – Role Based access control – SQL Injection – Statistical Databases security – Flow control – Encryption and Public Key infrastructures – Challenges

TOTAL: 45 PERIODS

Relational DatabaseDatabase

- ⇒ Database is a collection of data in a systematic way so that data can be accessed, managed and updated in an effective manner.
- ⇒ DBMS is a software, helps the user to interact with the data to analyze and use the data for application.
- ⇒ DB is developed by to manage the computerized data stored in a software.
- ⇒ In earlier days, DB helps to manage computerized data for a commercial application.
- ⇒ It plays an important role in banking, business, education and so on.
- ⇒ Handling of vast data and managing the business needs or data is the main purpose of DBS.
- ⇒ Security in data handling is a key role of DBMS.

Advantages of DBMS

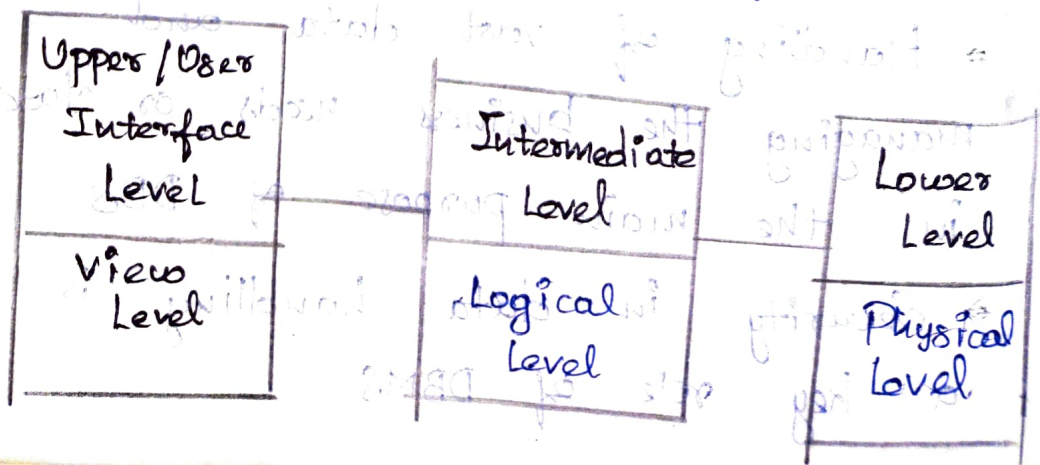
- *. Retenary Control*
- *. Efficient Query Processing
- *. Security from unauthorized access
- *. Persistent Storage
- *. Reduced Time
- *. Backup and recovery
- *. Multiple User Interface
- *. Updated Information
- *. Flexibility

Different Uses of Data

DBS gives a abstract view about data. Abstract view shows the external view of data in the uses of their utility Purpose and hide the internal information Such as how data formed, maintained and so on.

Data Abstraction

Data Abstraction defined as hiding internal information about data to the user. Internal information includes how data are stored, managed and so on aligned.



Based on hiding internal information, developers explain abstraction at three levels

- ⇒ Physical Level
- ⇒ Logical level
- ⇒ View level

Physical Level

It is the lower level of data abstraction describes about data structure and explain how data can be stored in DB.

Logical Level

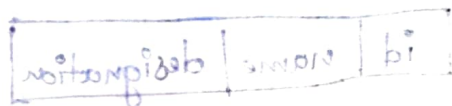
⇒ It is the middle level of the data abstraction, it describes what data to be shown. The Administrator decides what information on DB.

⇒ This level describes the relation among data structure and data used in middle level

View Level

* This is the user interface where describes only the part of entire database

* This level helps the user to access the data from the database



Eg

type employee record;

employee-id : int

employee-name : String

employee-salary : Double

end;

Instances

- * Instance is defined as a single occurrence or record of a particular entity in a database.
- * The collection of information stored in a database at a moment is called instance.
- * The data in the database which is being entered at a particular time is called State of a DB.

Scheme

The overall design of the entire description is called database Scheme.

Scheme diagram:

Employee

id	name	designation
----	------	-------------

Department

Code	exp	Salary
------	-----	--------

Grade

Seniors	Assistant	HR
---------	-----------	----

The concept of database scheme corresponds to the variable declared in the program like each variable has a particular value at a given instance. Object like employee, department, grade are called as scheme construction.

Scheme classification

* Physical schema

* Logical schema

* Subschema

Physical Schema

Database designed at physical level

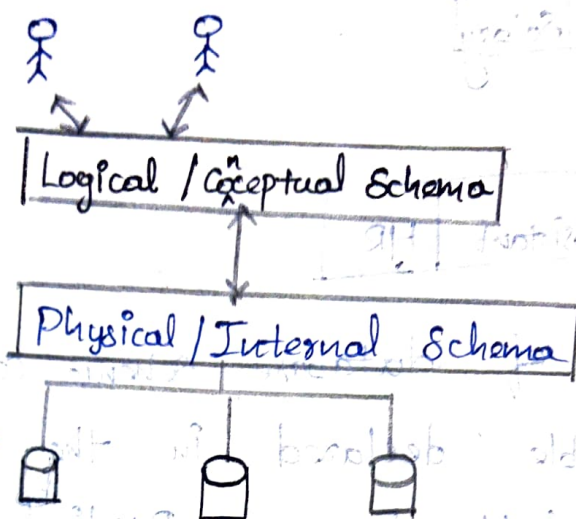
Logical Schema

Database designed at logical level

Three Scheme Architecture

Universal database MS from IBM uses relational model to describe a conceptual scheme gives an object oriented model to

describe the external schema



Data Independency in 3 Schema Architecture

Data Independency is defined as ability to change the schema gives a object oriented model to describe the external schema

Types

1. Logical data independency
2. Physical data independency

1] Ability to change the conceptual schema without changing on external schema. Conceptual schema can be changed the constant

2] Ability to change the internal schema without having change the conceptual schema

Data Model

* Data model is a collection of concept

describe the structure of database to understand the means of abstraction.

- * Data model describe a design of a database at different level of abstraction. Such as Physical, logical and view level.

- * Data constraints, Schematic that holds the data it also includes operations to specify as data updates and retrieval.

Eg:

It compute the employee salary, user-defined operation and database object, and generic operation perform like insert, delete, modify and basic model operations.

Four categories of data models are.

- * Relational

- * Entity relationship

- * Object based

- * Semi structured

Relational Model

Relation is a table of values of records. It means each row in a table have related to data value. The relational model has collection of table that represent the relationship among the data in a table.

Each table has row and column. Each Column has unique name. It is also known as record based model.

S.No	Name	Marks
1	Abi	99
2	Asrap	95

- * It is called record based since each table describes the info about particular record.
- * It is the basic model comprises of no. of fields and attributes.

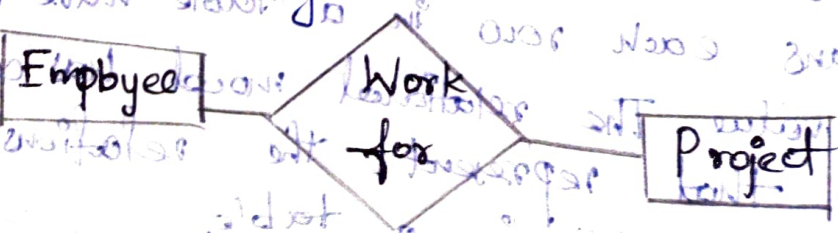
* The attributes refer to the column of table and tuple refers to the row of the table.

Entity Relationship

It is a popular high level conceptual data model. It is based on the real world where base of object is called entities.

Attributes and relationship

- ⇒ A entity is a real world entity. Eg. Person or object.
- ⇒ Array Attributes represent describe entity.
- ⇒ A relationship is the association among two or more entity.



Object based

- * It is an extending model with encapsulation, methods and object entity
- * It combines features of both object oriented and relational data model.

Database System Architecture

It is the component of database system and connection among them.

Database can be centralized (or) Client-Server which can be worked on the parallel computers Architecture

There are 3 level of Architecture

- 1] Internal Level
- 2] External Level
- 3] Conceptual Level

Internal Level

Internal level is a storage level and consider on what way the data stored inside the system

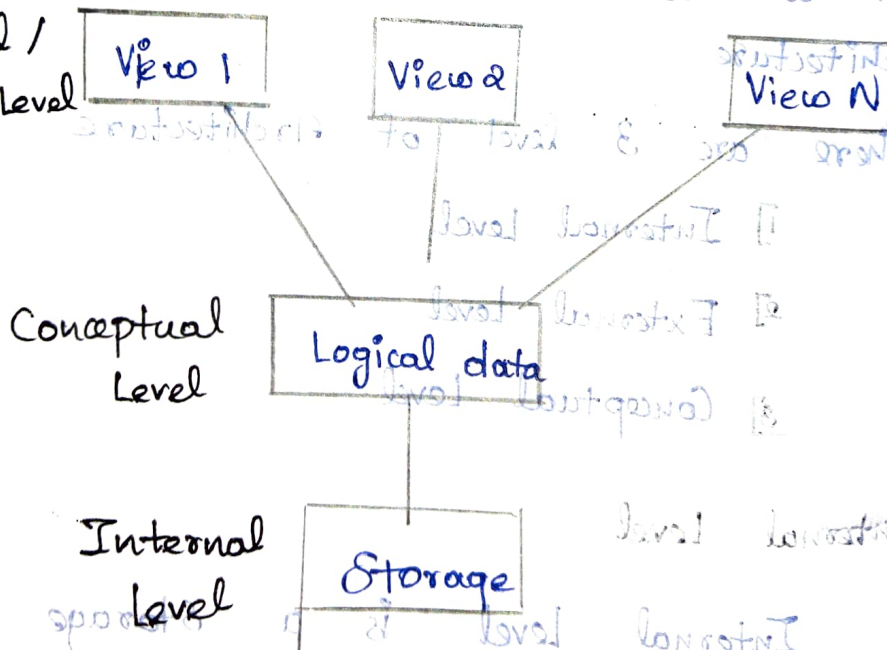
External Level

External Level is a user level
Consider about how data being fetched
and used by the user

Conceptual Level

Conceptual Level is a logical level
which acts as the middleware between the
Internal Level and External Level. It
Performs all logical level of computation
to maintain the data at the database

External /
View Level



Eg. of 3 Level

External - Struct student {

Int roll no;
char name[10];

float marks;

Conceptual Level

Student

Student number	int [5]
name	Character [15]
mark	Decimal [5]

Internal Level

Student

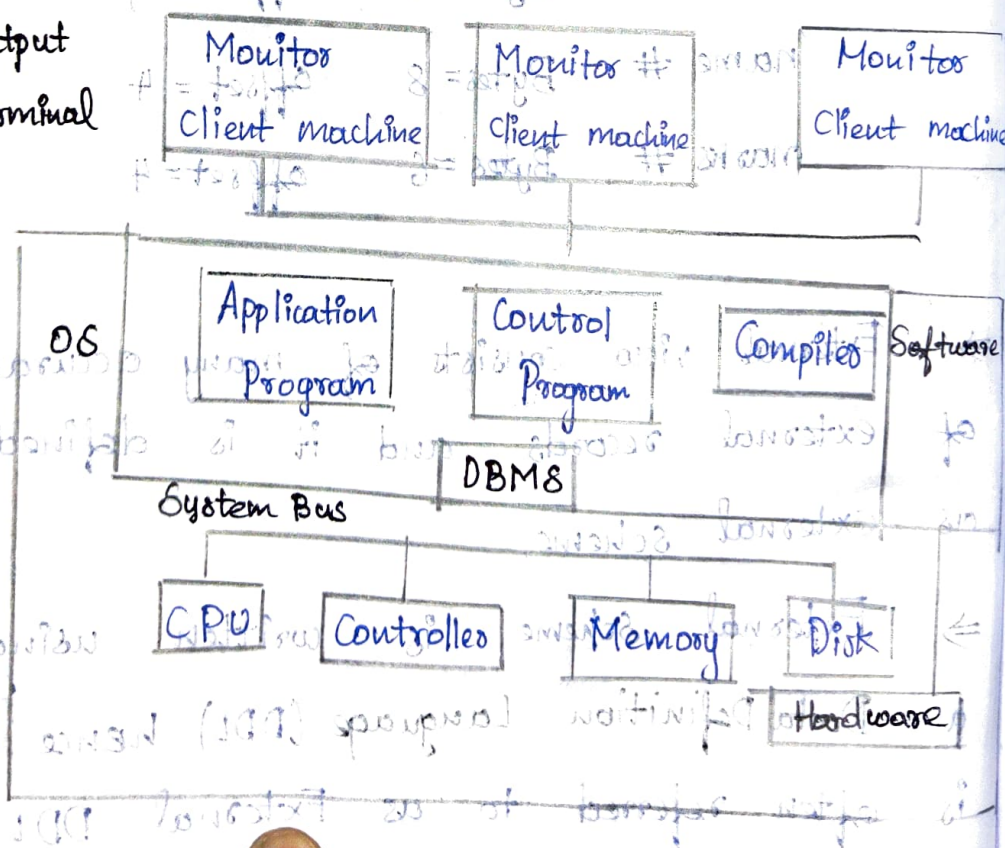
number #	Bytes = 6	offset = 6, Index num
name #	Bytes = 8	offset = 4
marks #	Bytes = 6	offset = 4

⇒ External view consists of many occurrence of external records and it is defined as external scheme.

⇒ External Scheme is written using a Data Definition Language (DDL) hence it is often referred to as External DDL

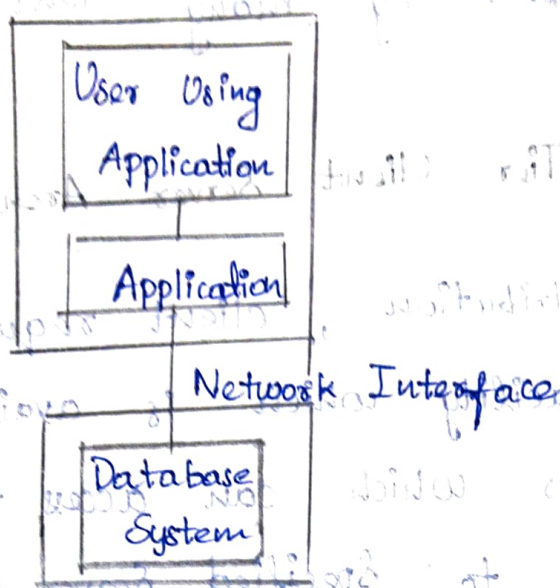
- ⇒ Conceptual View is a overall representation of an entire information content in the database, It is defined as conceptual Schema.
- ⇒ It also include additional features such as security and integrity constraints.
- ⇒ Internal View is a low level representation of a database contain stored record of the database. It is described as Internal Schema.
- ⇒ Internal Scheme is written using DDL is referred to as Internal DDL.

Output
Terminal



Generalized DBMS Architecture

Two Tier DBMS Architecture



⇒ The applications is partitioned into components that decides as a client machine. The Interfaces like ODBC and JDBC used for interaction between client and server.

⇒ Centralized DBMS perform application Program execution, user interface, control of DBMS in a single machine. user access the database centralized single mission.

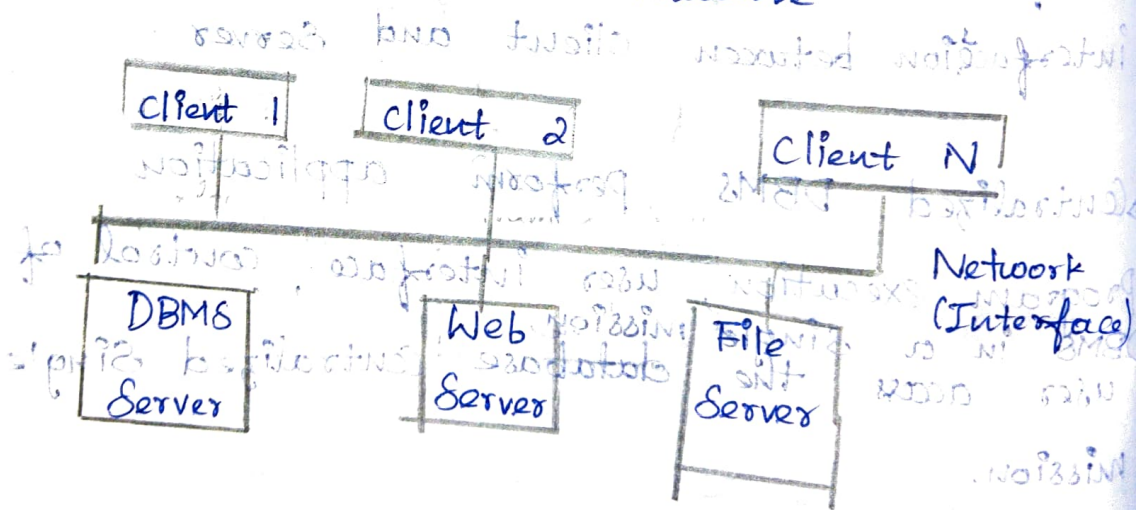
⇒ But in a client server architecture computing environment like Work station, file server, Database server, Web server connected to the network.

⇒ resources provided by the specific server can be accessed by many client machines

Logical Two Tier Client Server Architecture

⇒ In Client Distribution, client require additional facility which is available in another server which can access the service by connecting to specified server to the network

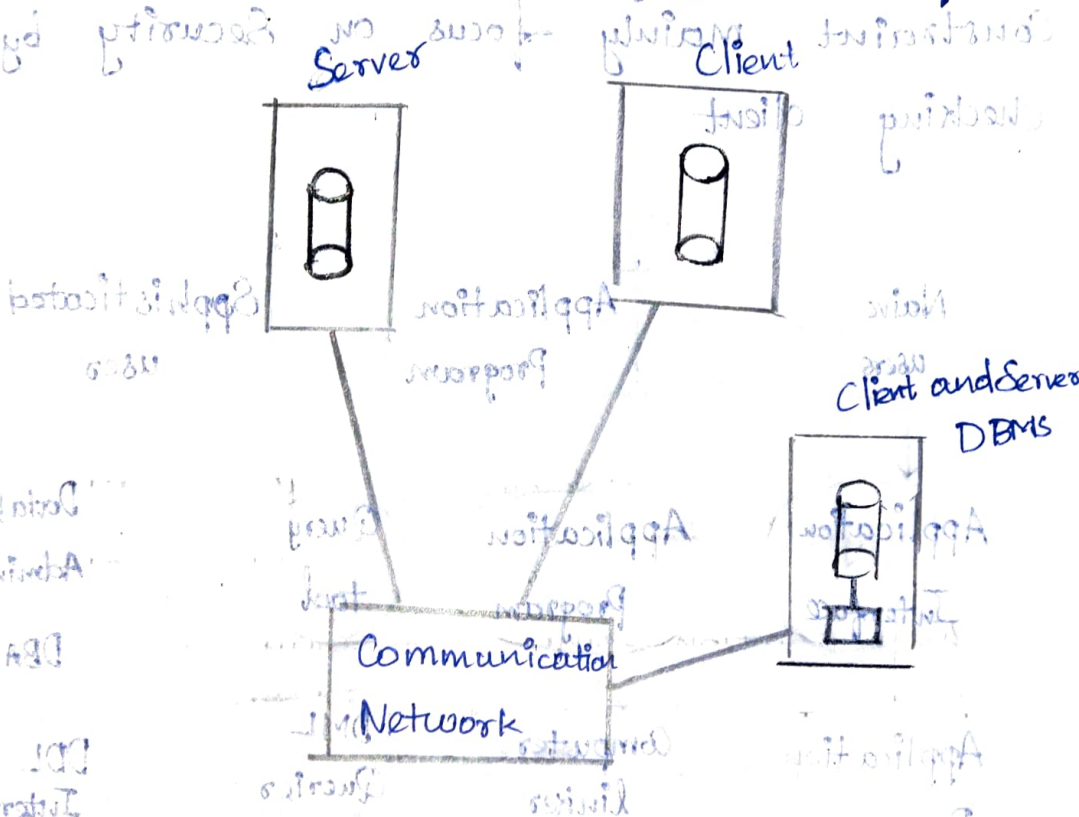
⇒ Server also a machine which holds the software and hardware support provide service to the client machine



Physical Two Tier Client Server Architecture

The machine would be dedicated server which is connected through the network

⇒ RDBMS, most of the machines started as centralized system. In a client server architecture the application program executed in client side when the client need to access the database which is on the server side. Then client start communicating through a program called open data base connectivity provided by application program interface.



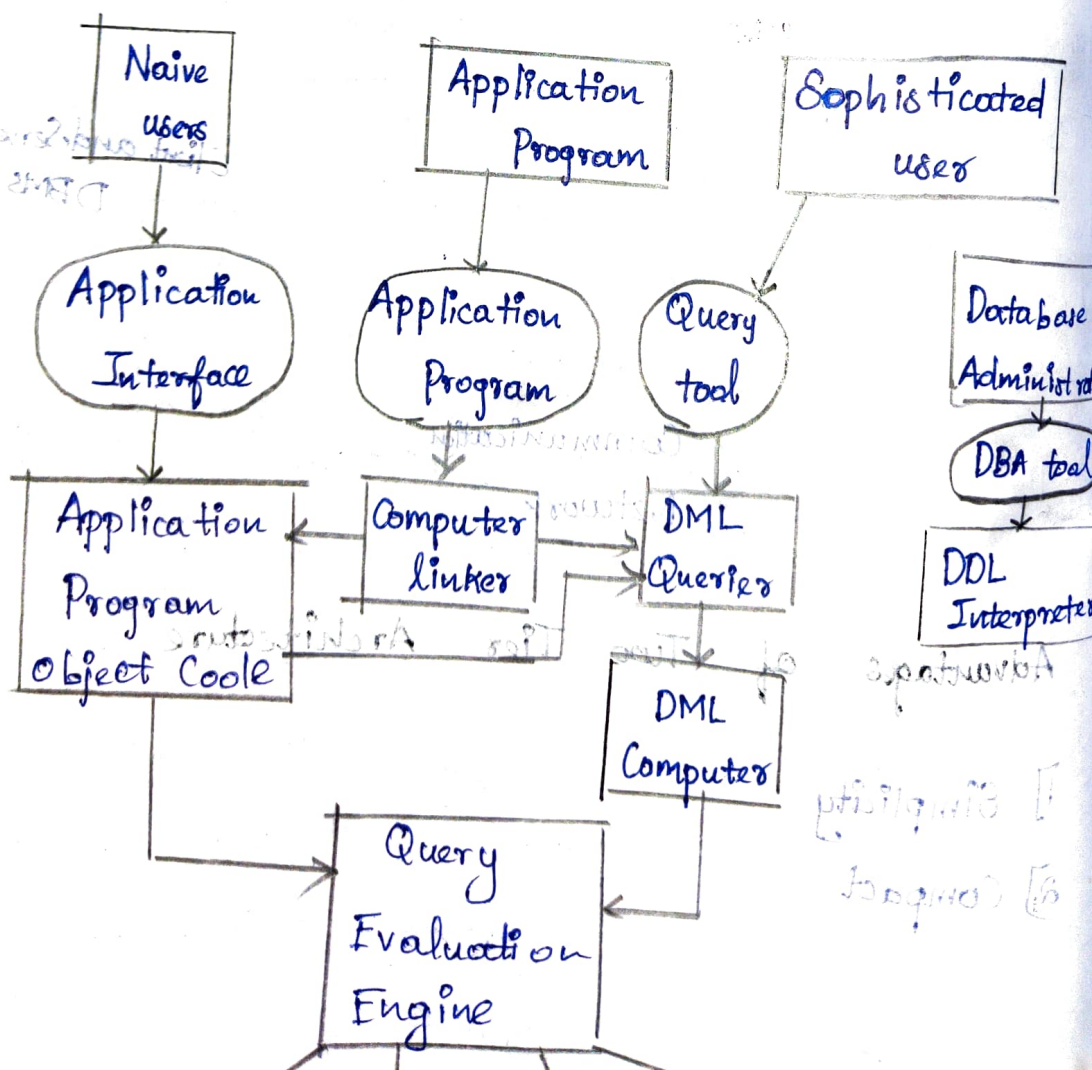
Advantages of Two Tier Architecture

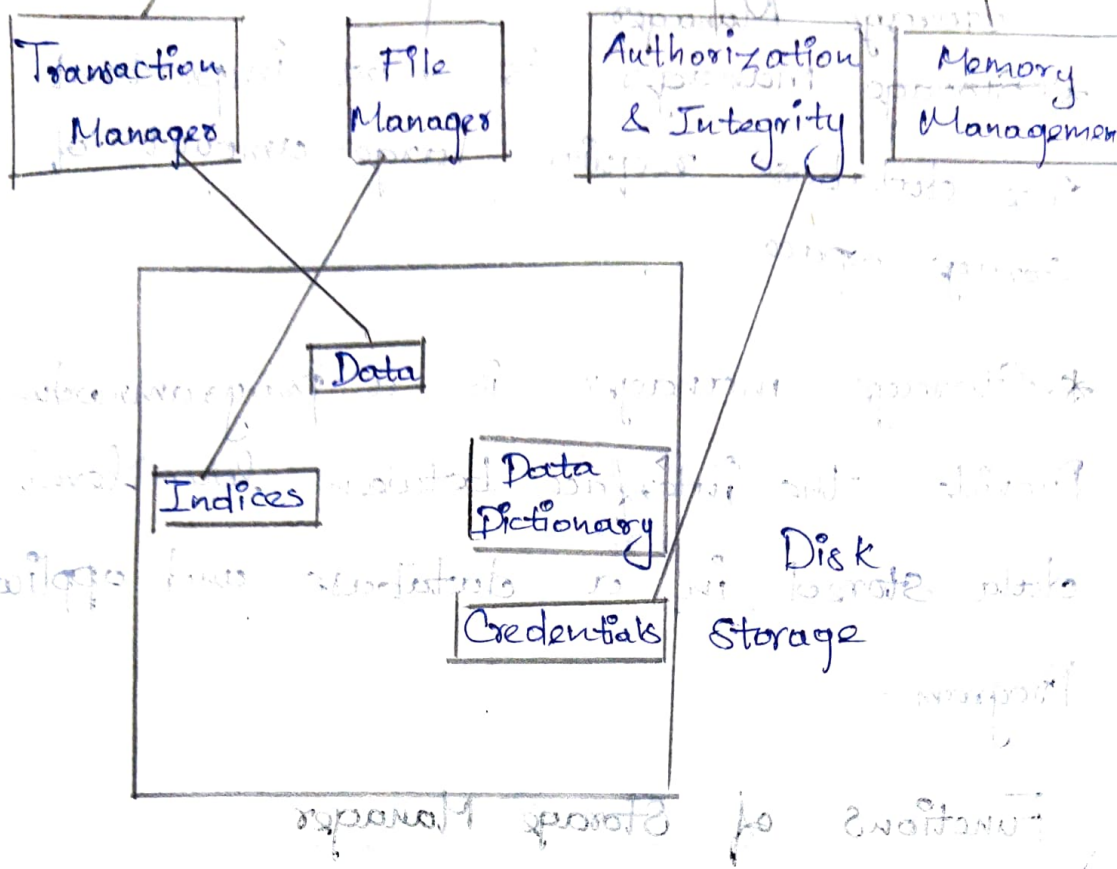
- 1] Simplicity
- 2] Compact

Three Tier Architecture

Emergence of web changed the role of Client and Server that needed the development of Three Tier Architecture

- ⇒ Middle Tier is Application / Web Server
- ⇒ Intermediate Server plays an intermediate role by storing business procedure and Constraint, mainly focus on Security by checking client





* Different Query processing Component
Plays a vital role to make the database function.

* Query processors fetch the require from the disk storage.

* This system structure focus on how to store a data, ensure atomicity, Perform concurrency control, distribute the Query Processing and so on.

* The functional component of the database is divided into storage manager and Query Processor.

Storage Manager

* Storage manager is most important
Since database require large amount of
Storage space.

* Storage manager is a program module
Provide the interface between low level
data stored in a database and application
Program.

Functions of Storage Manager

- * Interact with File Manager
- * Responsible for Storing, Retrieving and updating data in the database

Components of Storage Manager

- * Authorization and Integrity managers
- * Transaction Manager
- * File Manager
- * Buffer Manager

Query Processor

- * Query Processor perform DDL and DML Commands.
- * DDL Interpreter comprises of component
- * DML compiler to translate DML command in a query language to query evaluation engine.
- * Query evaluation engine execute low level instruction.

Relational Database

- * Relational database is a collection of data in terms of row and column, assigned unique name. row is called Tuple, column is called Attribute and the table is called a relation

- * A datatype format is also specified for each domain.

Atomic value

Each value in the domain is indivisible as far as the relational model is concerned.

Relational Schema (R)

* Relational Schema (R) denoted by $R(A_1, A_2, \dots, A_n)$ is made up of relation name R with list of Attributes $A_1, A_2, \dots, A_n$.

* Each Attribute role is play a some domain.

Characteristics of relation

- * Ordering of Tuple in a relation
- * Ordering of values with the tuple, values and NULL in the tuple.
- * A Tuple is a set of Pair i.e.
($\langle \text{attribute} \rangle, \langle \text{values} \rangle$)

Keys

- * Key is used to identify any record or tuple or attribute in a relation.
- * Key can be a Single Attribute or Group of attribute.
- * Key hold a unique value.

* Keys are used to fetch a particular data from the huge volume of relation.

Types of Key:

- 1] Super key
- 2] Primary key
- 3] Composite key
- 4] Candidate key
- 5] Secondary key
- 6] Non key Attribute
- 7] Non primary Attribute

Super key

Super key is a super set of Candidate key.

Candidate key

It refers to minimal set of fields uniquely identify the record in a relation.

Primary key

It is main key of any table that can uniquely identify a record in a table.

Composite key

It has more than one attribute group to uniquely identify a record in a relation table is called Composite key

Foreign key

It is a field in a table that identifies a row in another table

Secondary key

The candidate key which is not selected as a primary key is called Secondary key

Relational Algebra

Select, Project, union, difference, Cartesian product, rename, Set insertion, natural join, division, assignment

Emp Id	Emp-name	emp-dept	emp-phoneno
1.	AA	IT	1234
2.	BB	CSE	5678
3.	CC	EEE	9101
4.	DD	ECE	1234

Unary Operation

$\sigma \langle \text{Selection Condition} \rangle R$

$\sigma \langle \text{dept} = \text{IT} \rangle$

Output: AA
1234
IT

Projection Operation

emp-ID	emp_address
1	XX
2	YY
3	ZZ
4	SS

$\pi \langle \text{attribute} \rangle R$

Rename Operation

$\rho_s (B_1, B_2, \dots, B_n) (R)$

Binary relation Operation

- 1] Natural Join operation
- 2] Division operation
- 3] Assignment operation

Natural Join Operations

Natural Join expects a common column along different relation whereas it is not applicable in case of Cartesian Product. Join Symbol '⋈'

Inner Join

Theta Join, Equi join, Natural Join
are called Inner join.

Outer Join

Left Outer Join, Right Outer Join
Full outer join

Course	
A	B
100	DB
101	OS
102	DM

Head	
100	aaa
101	bbb
103	ccc

Left Outer Join

Course		Head	
100	DB	100	aaa
101	OS	NULL	NULL
102	DM	102	bbb

Right Outer Join

Course		Head	
100	DB	100	aaa
102	DM	102	bbb
NULL	NULL	103	ccc

Full Outer Join

Courses		IX	Head
A	B	C	D
100	DB	100	aaa
101	OS	NULL	NULL
102	DM	102	bbb
NULL	NULL	103	ccc

SQL Fundamentals

* Structure Query Language (SQL) is one of the fundamental language in database architecture.

* It is domain specific language used to manage the database.

* SQL has several parts

1] Data Definition Language

2] Data Manipulation Language

3] Data Control Language

4] Embedded SQL

5] Authorisation

6] Dynamic SQL

7] Transaction Control

8] View definition

1. Data Definition Language

Create, alter, drop, truncate, desc
Eg,

1. create table employee

(emp_no number(5), emp_name varchar(30));

2. insert into employee values (1, aa);

3. alter table employee add (dept varchar(10));

4. drop table employee;

5. delete from employee where emp_no: 2;

6. desc employee;

Output:

Name	Type
emp_no	number[5]
emp_name	varchar[30]

truncate is used to delete the entire

2. Data Manipulation Language

insert, update, rename

1. insert into employee values(1, 'IT', chennai);
2. update employee set exp=2 where emp_no=2;
3. rename employee to emp;

3. Data Control Language

grant and revoke

Eg,

SQL > Grant all on employee to individual

SQL > Revoke all on employee from individual

4. Transaction Control Language

- * 'Commit' it says the task completed
- * 'Savepoint' help to go back to previous transaction
- * 'rollback' restore the database upto last committed place.
- * 'Set Transaction' changes the transaction point

Implementation of Built in functions in SQL

1. upper - changes the character into uppercase
2. Lower - changes the character into lowercase
3. initcap
4. Length
5. Rpad
6. Lpad

7. translate

8. replace

9. trim

9. trim

10. 'datatype' function

the datatype.

11. "view definition" function

'virtual table' obtained based on Query

1] Updatable View

2] Readonly View

Eg,

1. Create view customer as (select branch-name, customer-name, from loan account where loan account-no = account, account-no)

Advanced SQL

User defined is two forms namely,

1. distinct type

2. Structure datatype

Large Object datatype

lob, clob - character data, blob - binary data

Catalog

Schema in turn contain data object and relationship among the data.

Integrity Constraint

* It is important to ensure the originality of data in the database

Eg, Every student in a student relation must be same as a student name in a formative test relation.

Constraining table creation

Normal Unique (check NULL is defined as domain for every attribute

Roll No numeric(10) not null;

Unique

Unique values of attribute form a candidate key. Primary key always

have a unique constrain

Eg, Create table Student (rollno int not null unique, name char[33] not null)

* Primary key constrain is used to identify each record in a relation

Eg, Create table student (rollno int unique, name char[33], Primary key (rollno))

* Primary key of one relation is a foreign key of Another relation.

⇒ check constrain is used to ensure attribute value are in a specific condition

Eg, Create table employee (emp-no int not null, empname char2[50], salary-number[20], check (salary > 0))

Set default

⇒ Set Default constrain is used to insert a default value in the attribute

Eg, Create table employee (emp-no int not null, emp-name char2[50], salary-number[20], default 2000)

Referential Integrity

Condition of set of tuples in one relation appear in certain set of tuples in another relation which is termed as Referential Integrity

Eg

1. create table dept (dept-id int not null, dept-name char[255], Primarykey (dept-id));
2. Create table employee (emp-id int not null, emp-name char[255], dept-id int foreign key (dept-id) reference department (dept-id) on delete cascade);

⇒ Assertion

Assertion is a statement in SQL that always expect to be exist in a database

Eg, create assertion balance check (select * from customer, where balance < 1000);

Authorization

Authorization to read, insert, create, update, delete data. In SQL DDL include grant and revoke privileges.

Embedded SQL

* It is way of combining programming language and SQL to support an application

* It is a method of inserting inline SQL queries into the programming language

* Embedded SQL is supported by Oracle database and not supported by MySQL.

Embedded SQL compilation is done in two step process.

1] SQL Statement is precompiled. It checks for the correctness and execution plan.

2] After precompilation the executable code is embedded into C, then C compiler compiles and executes.

Syntax:

```
EXEC SQL CONNECT database_name;  
EXEC SQL CONNECT user_name;
```

Host Variable

Host Variable captures the value passed by the query.

```
EXEC SQL BEGIN DECLARE SECTION
```

```
int empno;
```

```
char emp_name;
```

```
char dept[10];
```

EXEC SQL END DECLARE SECTION

Indicator Variable

This variable are two byte short type capture a NULL value

Two types of Embedded SQL

1. Static Embedded SQL

2. Dynamic Embedded SQL

Static Embedded SQL

* Entire SQL Statement is executed

during the precompilation process

* Host variable are not allowed to the

table name or attribute

Dynamic Embedded SQL

Entire SQL Statement is made to execute at run time using a Host

Variable

Dynamic SQL

Dynamic SQL is a SQL pre-processor during the run time.

Important features of SQL Server

- ⇒ In Static SQL it will compile at compile time and execute at run time.
- ⇒ In Dynamic SQL both compilation and execution is done at the run time.
- ⇒ In a Dynamic SQL till runtime we don't know the file-name, rows and columns. etc.

Advantages of Dynamic SQL

- * Rescue code for different dynamically built table.
- * Allow parameters filtering within clause.
- * Sorting by a column using conditional statement.

Disadvantages

- * Performance Loss
- * Temporary table cannot be accessed and it is declared as global.
- * Error Management is Time.

- * Maintenance is difficult.
- * Complex query tend to increase the execution time
- * SQL injection attacker Possible

DB Embedded SQL and Dynamic SQL

Embedded SQL	Dynamic SQL
The SQL Statement could not change at run time	The SQL Statement are constructed during run time
Since this statements are Static it can hard coded into an application	This statement built in during the run time
SQL database is pre determined to the SQL Query	Database is determined during the runtime
It is not flexible to the General purpose SQL Queries	Flexible to the General Purpose SQL Queries
SQL statements are compiled during compile time	SQL statements are compiled during run-time
It is less efficient since it is possible to access for the predetermined database	It is more efficient since it is used for general application
Here data is uniformly distributed	Here data is Non Uniformly distributed

It make use of
Execute immediate.
Execute and prepare.
Statements are not
available

It make use of
Execute immediate
Execute and prepare
statement

1976, January 1 to 1976, September 1

1985, January

1902, (contd.)

1. 1942-1943 1942 1941
 2. 1942-1943 1942 1941

11-20-1919

Study conducted, and
with patients of

6000

2011

[Faint handwritten notes at the bottom of the page]

of the same

1978-11-14

1900
 1901
 1902
 1903
 1904
 1905
 1906
 1907
 1908
 1909
 1910
 1911
 1912
 1913
 1914
 1915
 1916
 1917
 1918
 1919
 1920
 1921
 1922
 1923
 1924
 1925
 1926
 1927
 1928
 1929
 1930
 1931
 1932
 1933
 1934
 1935
 1936
 1937
 1938
 1939
 1940
 1941
 1942
 1943
 1944
 1945
 1946
 1947
 1948
 1949
 1950
 1951
 1952
 1953
 1954
 1955
 1956
 1957
 1958
 1959
 1960
 1961
 1962
 1963
 1964
 1965
 1966
 1967
 1968
 1969
 1970
 1971
 1972
 1973
 1974
 1975
 1976
 1977
 1978
 1979
 1980
 1981
 1982
 1983
 1984
 1985
 1986
 1987
 1988
 1989
 1990
 1991
 1992
 1993
 1994
 1995
 1996
 1997
 1998
 1999
 2000
 2001
 2002
 2003
 2004
 2005
 2006
 2007
 2008
 2009
 2010
 2011
 2012
 2013
 2014
 2015
 2016
 2017
 2018
 2019
 2020
 2021
 2022
 2023
 2024
 2025
 2026
 2027
 2028
 2029
 2030
 2031
 2032
 2033
 2034
 2035
 2036
 2037
 2038
 2039
 2040
 2041
 2042
 2043
 2044
 2045
 2046
 2047
 2048
 2049
 2050
 2051
 2052
 2053
 2054
 2055
 2056
 2057
 2058
 2059
 2060
 2061
 2062
 2063
 2064
 2065
 2066
 2067
 2068
 2069
 2070
 2071
 2072
 2073
 2074
 2075
 2076
 2077
 2078
 2079
 2080
 2081
 2082
 2083
 2084
 2085
 2086
 2087
 2088
 2089
 2090
 2091
 2092
 2093
 2094
 2095
 2096
 2097
 2098
 2099
 2100
 2101
 2102
 2103
 2104
 2105
 2106
 2107
 2108
 2109
 2110
 2111
 2112
 2113
 2114
 2115
 2116
 2117
 2118
 2119
 2120
 2121
 2122
 2123
 2124
 2125
 2126
 2127
 2128
 2129
 2130
 2131
 2132
 2133
 2134
 2135
 2136
 2137
 2138
 2139
 2140
 2141
 2142
 2143
 2144
 2145
 2146
 2147
 2148
 2149
 2150
 2151
 2152
 2153
 2154
 2155
 2156
 2157
 2158
 2159
 2160
 2161
 2162
 2163
 2164
 2165
 2166
 2167
 2168
 2169
 2170
 2171
 2172
 2173
 2174
 2175
 2176
 2177
 2178
 2179
 2180
 2181
 2182
 2183
 2184
 2185
 2186
 2187
 2188
 2189
 2190
 2191
 2192
 2193
 2194
 2195
 2196
 2197
 2198
 2199
 2200
 2201
 2202
 2203
 2204
 2205
 2206
 2207
 2208
 2209
 2210
 2211
 2212
 2213
 2214
 2215
 2216
 2217
 2218
 2219
 2220
 2221
 2222
 2223
 2224
 2225
 2226
 2227
 2228
 2229
 2230
 2231
 2232
 2233
 2234
 2235
 2236
 2237
 2238
 2239
 2240
 2241
 2242
 2243
 2244
 2245
 2246
 2247
 2248
 2249
 2250
 2251
 2252
 2253
 2254
 2255
 2256
 2257
 2258
 2259
 2260
 2261
 2262
 2263
 2264
 2265
 2266
 2267
 2268
 2269
 2270
 2271
 2272
 2273
 2274
 2275
 2276
 2277
 2278
 2279
 2280
 2281
 2282
 2283
 2284
 2285
 2286
 2287
 2288
 2289
 2290
 2291
 2292
 2293
 2294
 2295
 2296
 2297
 2298
 2299
 2300
 2301
 2302
 2303
 2304
 2305
 2306
 2307
 2308
 2309
 2310
 2311
 2312
 2313
 2314
 2315
 2316
 2317
 2318
 2319
 2320
 2321
 2322
 2323
 2324
 2325
 2326
 2327
 2328
 2329
 2330
 2331
 2332
 2333
 2334
 2335
 2336
 2337
 2338
 2339
 2340
 2341
 2342
 2343
 2344
 2345
 2346
 2347
 2348
 2349
 2350
 2351
 2352
 2353
 2354

[illegible]

200 The volume 179,
with one private collection.

2000 10/10/2000

bright some of it
 as, however, it is
 admitted among

5507 to 5511 431 2

7055 to 7060 431 2

7061 to 7065 431 2

What is the result?

1947-1948

Unit - 2

Database Design

* Database Design is a logical structure of a database. It is also called as Database model.

E-R Model (Entity - Relation)

* Entity is a real world object distinguished from the other object.

* A Entity is described using a set of attributes.

* Entity Set is a collection of similar entities.

Ex, All employees

* A Entity is a Entity Set having same set of attributes.

* Each entity has a key.

* Each attribute has a domain.

Relationship

* A Relationship is the association among the several entities.

* A relationship set is a mathematical relation among $n \geq 2$

* Attribute have a value of specific domain or type of column

* Using a key attribute other details of relationship can be identified

Types of Attribute

1] Simple and composite attribute

* Simple attribute cannot be divided into similar components

Ex. Age

* Composite attribute is the combination of different attribute. It can be divided into similar components

Ex, ~~Courses joined by Student~~, Date of birth

2] Single and Multivalued Attribute

⇒ Each Entity takes a single value for an instance

Ex. Age

In Multivalued Attribute, entity take many attribute.

Ex, Course joined by Student

3] Stored and Derived attribute

* Stored Attributes are Attributes which are needed to be stored permanently in the database

Ex. Name of the Student

* Derived Attribute are derived from other attribute. This attribute is obtained from manipulation of corresponding attribute

Ex. Age from a database

Descriptive Attribute

* Attribute comes with a relationship is termed as the descriptive attribute

Five degrees of relationship

* Unary

* Binary

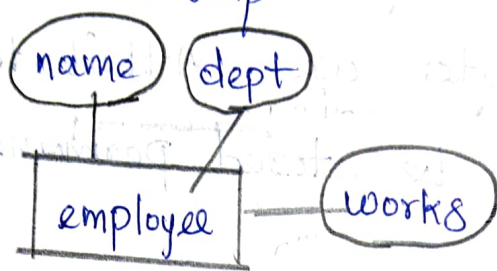
* Ternary

* Quadnary

* Nnary

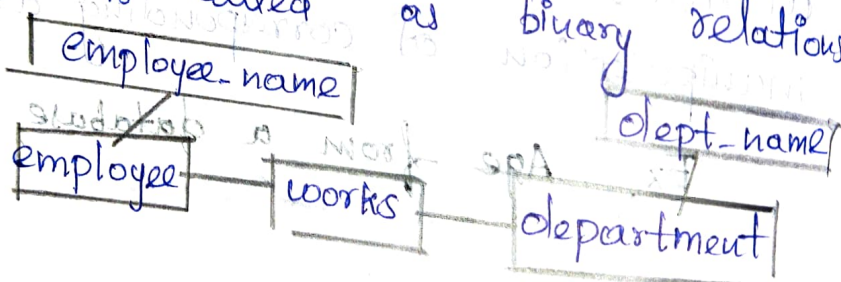
Unary relationship

* When a association of an attribute exist with a single entity is called as unary relationship



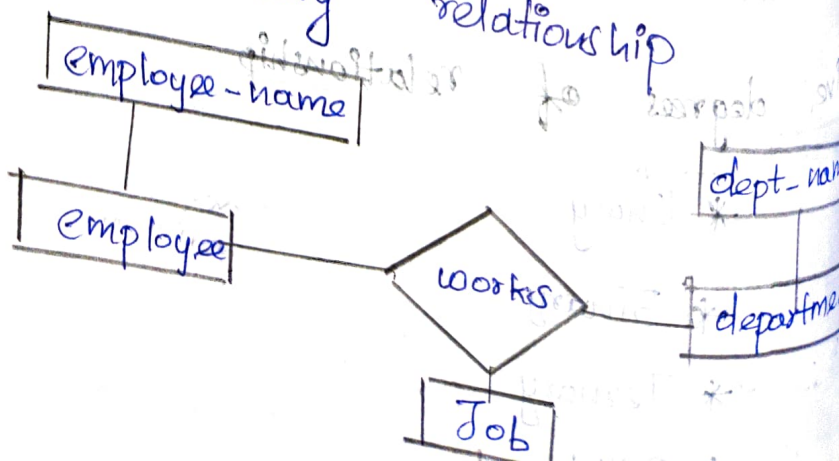
Binary relationship

When a two entity exist in a association which is called as binary relationship



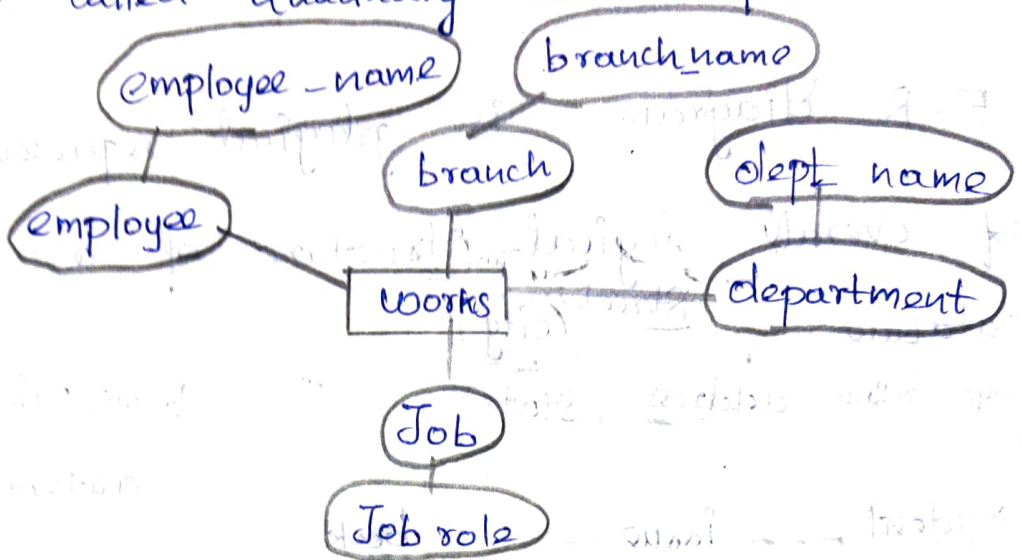
Ternary relationship

When a three entity takes place it called as Ternary relationship



Quadrary relationship

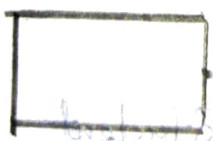
When a four entity takes place then it is called Quadrary relationship.



Nnary relationship

It is more than four entity exist then it is called Nnary relationship

Component of E-R diagram



→ entity



→ Specialization
Generalization



→ relationship



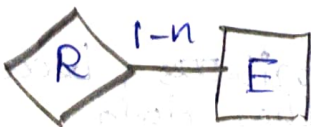
→ links



→ Attributes



→ Multivalued
attributes

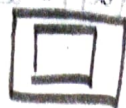


→ Cardinality
limits



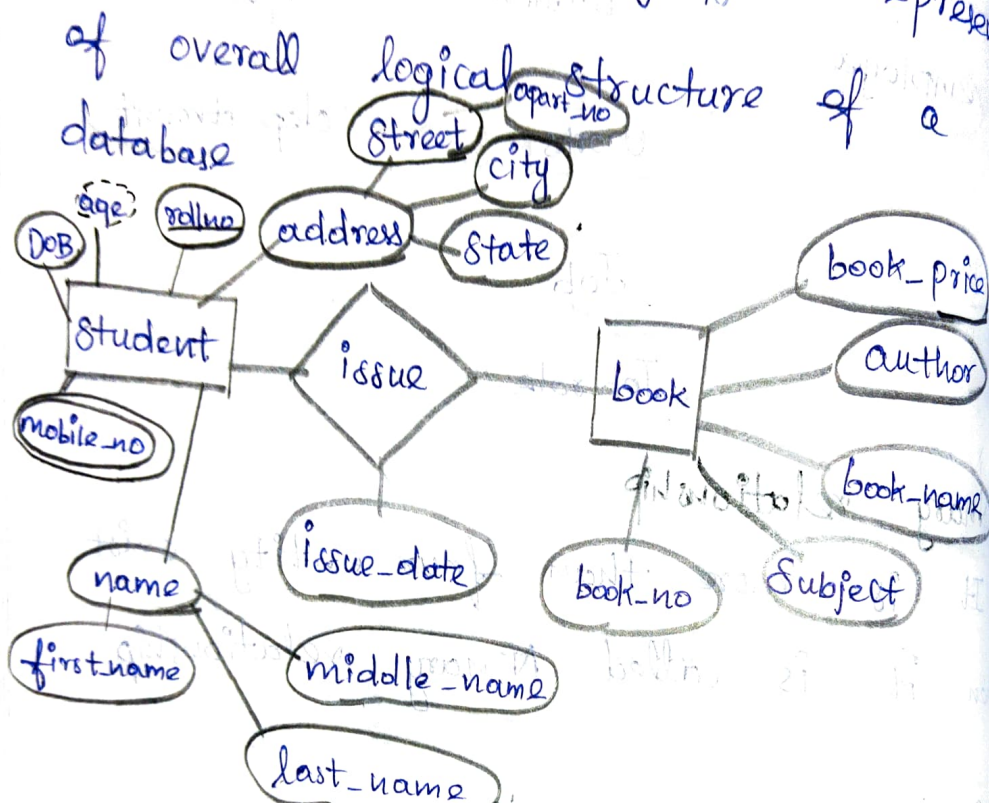
→ derived
attributes

= → Total Participation



Weak Set

E-R diagram is graphical representation of overall logical structure of a database



- * Student and book are entity
- * issue is a relationship between student and book
- * attributes in a entity are denoted by oval shape
- * mobile number is a multivalued attribute. One student can have more than one mobile number
- * age is a derived attribute because it is derived from other data

* rollno is a primary attribute

* name is a composite attribute which is decomposed into multiple attributes such as first, last and middle name

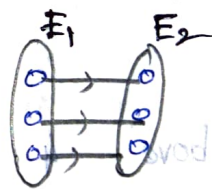
* address and street are composite attribute

* issue is a relationship which is descriptive attribute

Cardinalities Mapping

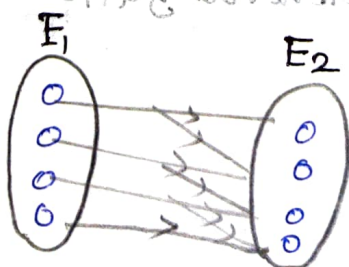
i] one to one relation

Here an entity e_1 is associated with atmost 1 entity in e_2

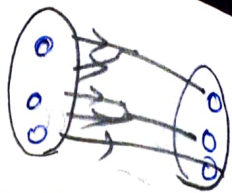


ii] One to many cardinality

An entity e_1 is associated with one of more than 1 entity in e_2 but e_2 must associated with 1 entity in e_1



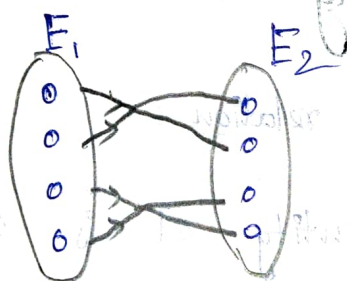
iii] Many to One



An entity E_1 is associated with atmost 1 entity in E_2 . E_2 must be associated with any no. of entity in

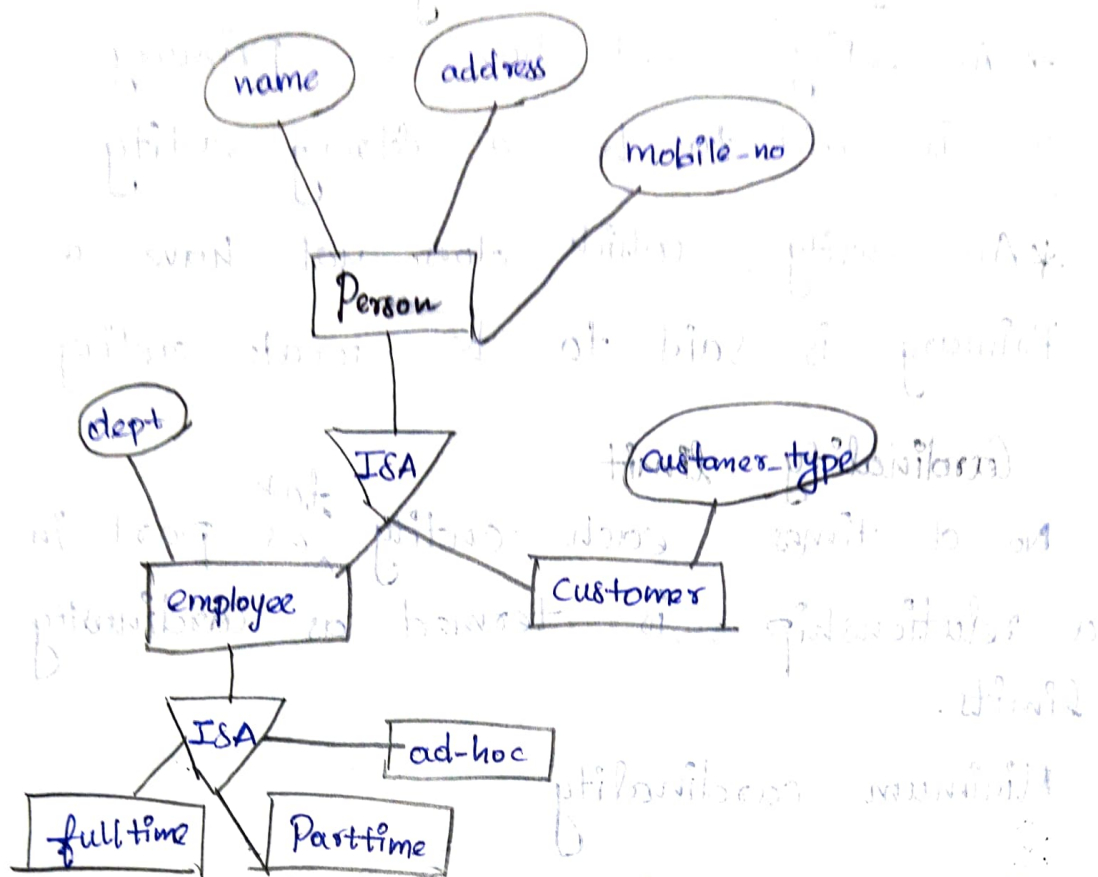
iv] Many to Many

An entity E_1 is associated with 0 or more no. of entities in E_2 and entity E_2 must be associated with or more no. of entities in E_2



Above 4 mapping constrain can be classified as total Participation constrain Partial Participation constrain.

Specialization and Generalization of E - R Diagram



* Person entity is specialized as employee and customer entity

* employee is further specialized as fulltime, Part time and ad-hoc.

Binary Relationship

Here two entities are connected to a relationship then it is called Binary relationship.

Strong and Weak Entity

- * An entity which have a primary key is said to be a strong entity
- * An entity which does not have primary is said to be weak entity

Cardinality limit

No. of times each entity ^{take} as part in a relationship is termed as Cardinality limits.

Minimum cardinality

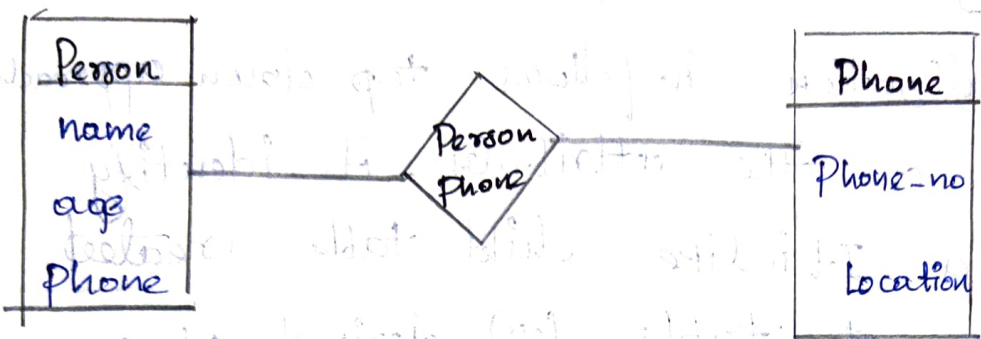
- * Zero no. of times the entity takes place in a relationship. total no. of Participation of an entity in a relationship.
- * Maximum value at most one time Participation of an entity in a relationship. n no. of times the entity

E-R model design issue

Use of entity set and Attribute

- * Consider an entity Person, attribute Phone number. That Person can have this phone number as office LAN number or Personal phone number. So we used

Create separate phone number entity to add an attribute phone no. and location



Use of Entity Set & Relationship Set

one entity set can be related into two or more different entity set.

Ex, Registration of course related to Student and course available



Binary & Non Binary

Non binary relationship can be explained by binary relationship. It is possible to have a distant, ^{Non binary} relationship set by a distant binary relationship set.

Placement of relationship attribute

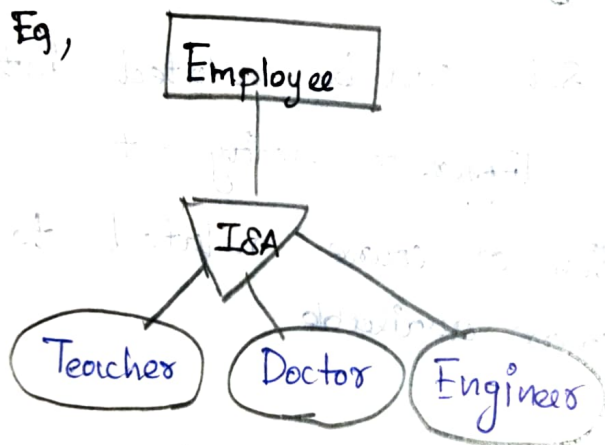
The design of placing descriptive attribute should reflect the characteristics of entity

been used by a model E-R diagram

Entity enhanced relationship

⇒ Specialization infollows top down approach based on the attributes it identifies entities. It is like child table created by parent table (or) derived class from the existing class table

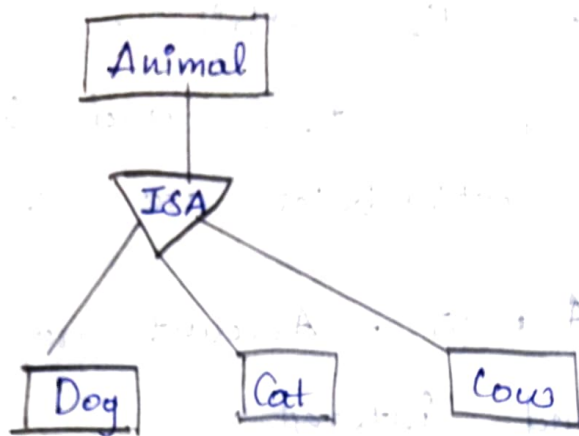
Eg,



* Generalization follows Bottom up approach

It works based on the child table finds the mother table. It finds base class from the derived class

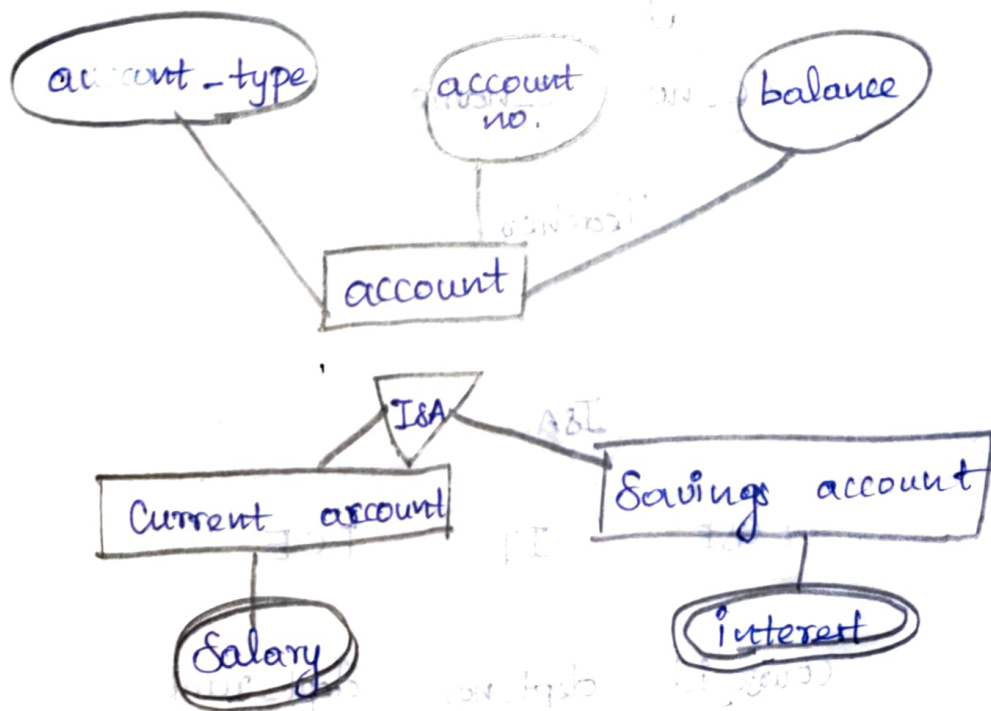
* Child entity take a attribute from the higher level of entity. It performs the multilevel inheritance in the entity set



Design a constrain on Specialization and Generalization.

Condition: Defined constrain

- * Lower level entities attributes are identified based on the condition satisfied by the upper level entities.
- * It can take single level inheritance or Multilevel inheritance.



⇒ Based upon the attributes such as account type, account_no and balance lower level attributes are identified.

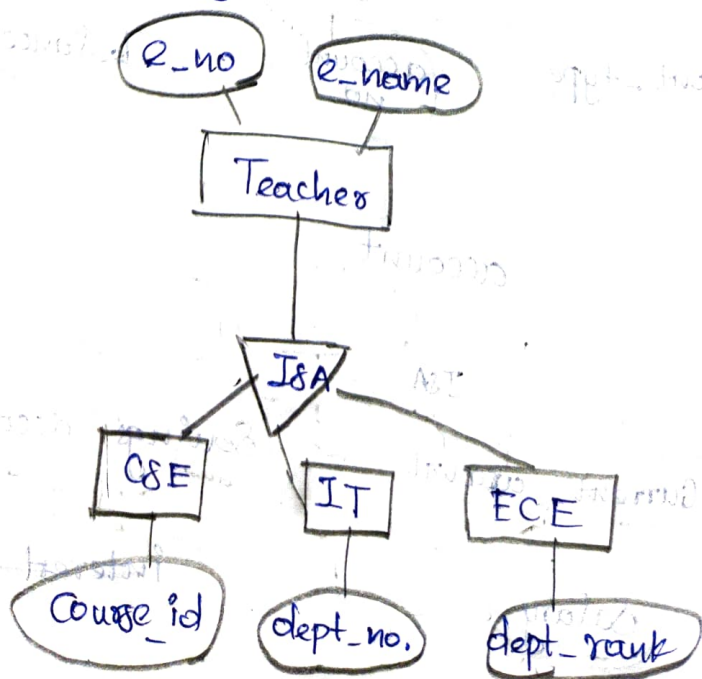
* Savings Account - Account no, account balance and interest

* Current Account - Account no, account balance and salary

User Defined Constraint

* based on the user level the lower level entities attributes are identified

* Constraint based on whether attribute of an entity belong to one entity or more than one entity.



Constraint based on completeness

It specifies occurrence of an each entity supertype may be at least member of one subtype.

Partial Completeness

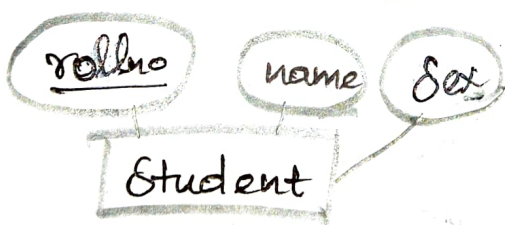
Not all the members of supertype will be members of subtype

Full Completeness

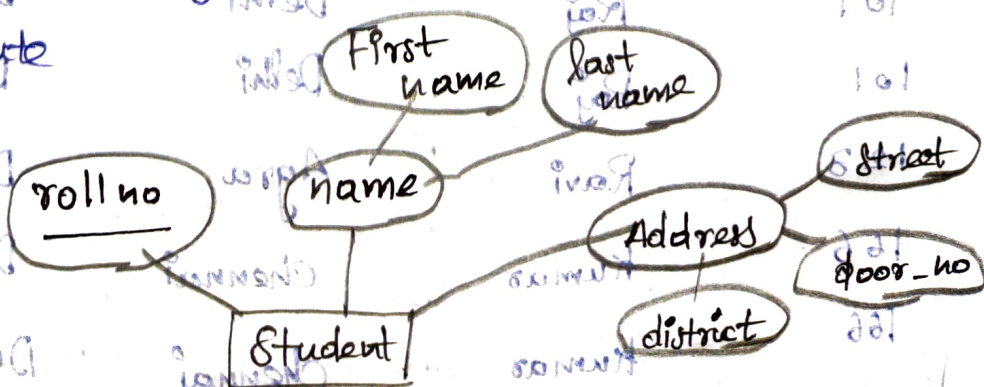
Every supertype occurrence must be a member of one subtype

E-R to relational mapping

⇒ Strong entity with single attribute



⇒ For a strong entity with composite attribute



* For a strong entity set with any no. of composite attribute will require only one table in a relational model. While conversion simple attribute of the composite attribute are taken into the account and not the composite attribute itself.

Need for Normalization

Anomalies in DBMS.

There are 3 types of anomalies that occur when the database is not normal.

These are insertion, updation and deletion anomaly.

Eg

Emp-id	Emp-name	Emp-address	Emp-dept
101	Raj	Delhi	D001
101	Raj	Delhi	D002
123	Ravi	Agra	D890
166	Kumar	Chennai	D900
166	Kumar	Chennai	D004

1] Update Anomaly

2] Insert Anomaly

3] Delete Anomaly

Properties of Normalization

i] Loseless decomposition

ii] Dependency preservation

i] When the decomposition of relation are into r_1 and r_2 produce the same relation r Joining r_1 and r_2 is called loseless decomposition.

Ex Student

Reg-No	Student-name	dept
1	Bala	cse
2	nandha	It
3	Pandi	cse

Reg-No	Student-name
1	Bala
2	Nandha
3	Pandi

Reg-No	dept
1	CSE
2	IT
3	CSE

ii] Dependency Preservation

⇒ The decomposition of relation r functional dependency $f = r_1, r_2$ to functional dependency f_1 and f_2 respectively, It is known as dependency preservation.

$R(ABCDEG)$

$F = \{ AB \rightarrow C \quad B \rightarrow D$

$AC \rightarrow B \quad BC \rightarrow A$

$AD \rightarrow E \quad E \rightarrow G \}$

Decompose into $R_1(AB)$, $R_2(BC)$, $R_3(ABDE)$ and $R_4(EG)$

First Normal Form (1NF)

- ⇒ All attributes (column) in the entity (table) must be single valued.
- ⇒ Repeating or multi valued attributes are moved into a separate entity (table).

Eg,

Customers

Cid	Name	Contact No
C01	aaa	{123456789}
C02	bbb	{123, 333, 456}
C03	ccc	

Soln:

Customers	
Cid	Name
C01	aaa
C02	bbb
C03	ccc

Customer Contact

Cid	Contact_No
C01	123456789
C02	123
C02	333
C02	456

The above tables are now in normalized form (1NF)

Properties of 1NF

- * Each attribute value must be atomic
- * Value in a attribute must be in a same domain
- * All attributes should have unique name

Second Normal Form (2NF)

- * Each entity must be 1NF
- * Each attribute within the entity must slowly depend on unique identifier of the entity

Properties of 2NF

- * It must be 1NF. It should not have a Partial dependency.

Consider the following relation

Teacher_id	Subject	Teacher_age
111	Maths	38
111	Physics	38
222	Biology	38
333	Physics	40
333	Chemistry	40

⇒ The above table is in 1NF, However it is not in 2NF, because Non-prime attribute (Teacher_age) is dependent on (Teacher_id) alone, which is a proper subset of candidate key.

⇒ To make the table satisfied 2NF, the relation is split into two tables like this

Teacher Details

Teacher Subject

Teacher_Id	Teacher_age
111	38
222	38
333	40

Teacher_id	Subject
111	Maths
111	Physics
222	Biology
333	Physics
333	Chemistry

Prime and Non-prime * attribute is part of
 a Candidate key is known as

The attribute is not part of a candidate key is known as Non-prime Attribute

Third Normal Form (3NF)

A table is said to be in 3NF, if it contains the following conditions.

- * It should be in 2NF

- * Transitive functional dependency should be removed. (In other words non prime attribute should be no dependent on Non-prime attribute.)

Transactions

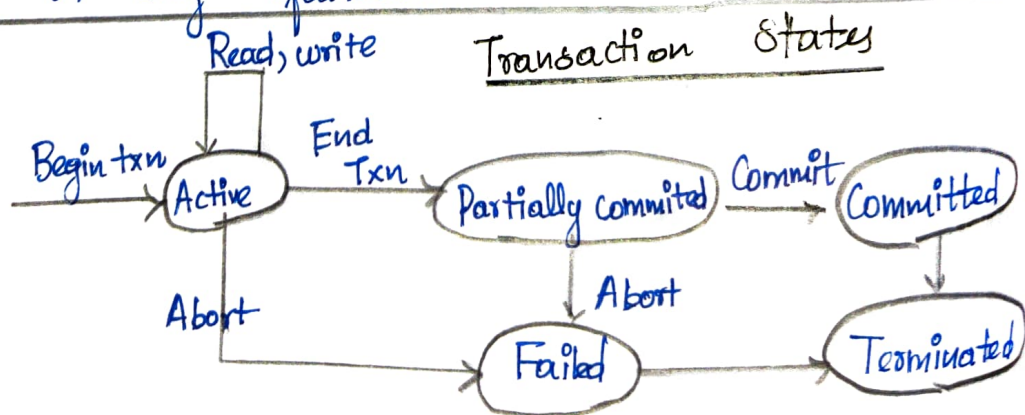
- * Transaction is defined as the group of task performed by a user or application
- * It is a set of changes executed together to update or read the content of the database.

Properties

- A - Atomicity (roll back, abort, commit)
- C - Consistency
- I - Isolation
- D - Durability

Durability

- * The changes applied to a database by a committed transaction should be permanently saved in database
- * The changes must not be lost because of any failure.



* A txn goes into an active state immediately after it starts execution, where it can issue READ and WRITE operations. When the txn ends it moves to the Partially committed state.

* At this point, some recovery protocols need to ensure that a system failure will not result in an inability to record the changes of the txns permanently. Once this check is successful, the txn is said to have reached its commit point and enters committed state.

* However, a txn can go to the failed state if one of the checks fails or if the txn is aborted during its active state.

Ex,

Show how ACID property can be achieved during a system failure in case of money transaction from A to B.

Soln: Solution is to maintain every executed task of T_i in A in a Volatile Storage, only when all the tasks have been

completed, commit can be performed in the same way T2. of B takes place.

Student		
Reg-No	Name	Marks
1	aaa	80
2	bbb	85
3	ccc	90

SQL > Begin

SQL > insert into student values (4, 'ddd', 90)

SQL > select * from student

SQL > commit

Schedules in Transactions

* A Schedule S for n transactions $T_1, T_2, \dots, T_n$, is ordering of a transaction ^{operation} in chronological order.

* To run a multiple transaction concurrently the scheduling of an instruction in transaction is termed as schedule.

Eg,

T_1	T_2
$R(x)$	
	$W(x)$

Characteristics of Schedule

- * Each transaction must have complete set of instructions.
- * Order of execution of instruction for a transaction must always remain same.
- * Schedule can be done serially or concurrently.

Different types of Schedule

- * Serial Schedule
- * Complete Schedule
- * Recoverable Schedule
- * Cascadeless Schedule
- * Strict Schedule

Serial schedule

The operation of one transaction does not interleaved the operation of other transaction.

Eg,	T_1	T_2
	$A = 50$	
	read(A)	
	$A = A + 50$	
	write(A)	
		read(A)
		$A = A + 50$
		write(A)

Recoverable Schedule

If there are two transactions T_1 and T_2
 If T_2 is accessing a data which is previously written by T_1 , Then the commit operation of T_1 must appear before the commit operation of T_2 for each pair of transaction

T_1	T_2
$A = 50$	
Read(A)	
$A = A + 50$	
Write(A)	
	Read(A)
	$A = A + 100$
	Write(A)
Read(B)	
Write(B)	
Commit	Commit

Cascadeless Schedule

If T_1 and T_2 are two transaction, T_2 access a data from T_1 then commit operation of T_1 must occur ^{before} the read operation of T_2

⇒ Cascading means effect of migration of one transaction to others.

cascading		Abort
T_1	T_2	T_3
Read ₁ (x) Write ₁ (x)	Read ₂ (x) Write ₂ (x)	Read ₃ (x) Write ₃ (x)
Abort	Abort	Abort

Cascadeless Schedule

T_1	T_2	T_3
Read ₁ (A) Write ₁ (A) Commit	Read ₂ (x) Write ₂ (x) Commit	Read ₃ (x) Write ₃ (x) commit

→ In General, committed transaction does not come under the concept of abort, because transaction is already done or over.

Strict Schedule

In cascadeless schedule, transaction T_2 must read from T_1 to T_2 must done after the commit in T_1 but there is no criteria for write() operation.

T_1	T_2
Read(x)	
Write(x)	
	Write(x)
Commit	Read(x)

Transaction Isolation levels in DBMS

Transaction isolation level is referred the degree at which one transaction is isolated from other transaction.

There are 10 factors determine

i] Dirty-Read

ii] Non Repeatable-read

iii] Phantom read

i] Dirty read

T ₁	T ₂
begin read(x)	begin read(x)

The activity of the read performed by T₂ before T₁ commit is called Dirty read

ii] Non repeatable read

T ₁	T ₂
1. begin	begin
2. x = 2	
3. read(x)	
4.	read(x)
5.	x = x + 2
6.	Commit
7. read(x)	

Phantom Read

When same queries executed more than once in a transaction then it is called Phantom read

Serializability

A schedule 'S' of 'n' txns is serializable if it is equivalent to some serial schedule of the 'n' txns. This property is called serializability.

T ₁	T ₂
R	R
W	W
R	W
W	R

There are 3 types of Schedule equivalences

1. Result equivalence
2. Conflict equivalence
3. View equivalence.

1. Result

If a same result is produced by a schedule then it is called result equivalence.

2. View

If two schedules perform a same set of operations in a similar manner, then it is referred to as view equivalence.

3. Conflict Equivalence

A pair of operations are said to be in conflict, if they satisfy the following conditions

- 1] Both operations belong to different ^{same} txns
- 2] They access the data
- 3] At least one of the operations is write operations

Eg $T_1: R(A), T_2: W(A)$

T_1	T_2
$R(A)$	$W(A)$

Conflict Serializable Schedule

A schedule is conflict serializable, if it is conflict equivalent to any of serial schedule.

View Serializable Schedule

A schedule is view serializable, if it is view equivalent to any serial schedule

Conditions

1. Initial read
2. Read / Write sequence
3. Final update

2 Phase Locking

1. Growing Phase (Locking Phase)
2. Shrinking Phase (Relief Phase)

1. Growing Phase

In this phase, new locks on the data item can be acquired but none can be released.

2. Shrinking Phase

In this phase, existing locks can be released, but no new locks can be acquired.

Lock

A Lock is a variable which is associated with a data item, describes the status of data item with respect to the operation applied to it.

2 Phase Locking System

2 Phase locking perform concurrency control.

Problems occurs due to 2PLock

1. Early Lock
2. Dead lock
3. Cascading rollback

Types of 2 Phase Locking

- 1] Strict Two phase locking
- 2] Rigorous Two phase locking

Shadow Paging (No Undo No Redo)

It is a recovery scheme in which data base is considered to be made up of ^{no. of} fixed size disk pages

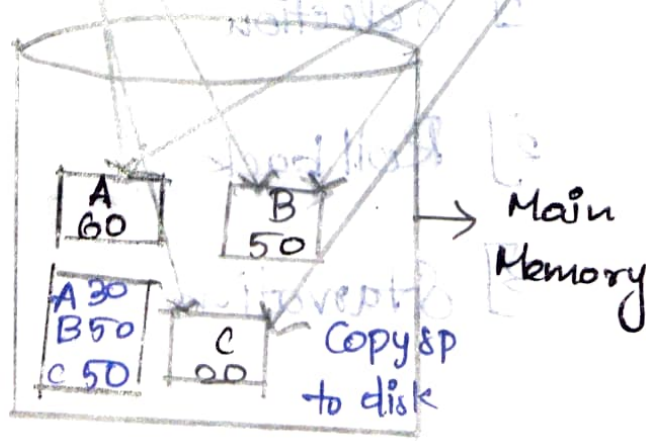
Structure:

B.No	Pointer
1	A 30
2	B 50
3	C 50

Current Page

B.No	Pointer
1	$A 60 - 30 = 30$
2	B 50
3	$C 20 + 30 = 50$

Shadow Page



Advantages

* Log file is not necessary

* No UNDO No REDO operation

Disadvantages

* The use of pointer is overhead

* Data fragmentation

Transaction Recovery

To ensure atomicity despite, we first output information describing the modifications to stable storage without modifying the database itself.

There are two approaches for recovery

1] Log based recovery

2] Shadow Paging

Deadlock Recovery

1] Selection

2] Roll back

3] Starvation

Write Ahead Logging Rule (WAL)

Approaches in Transaction Recovery

- 1] Deferred database modification
- 2] Immediate database modification

ARIES Algorithm

- ⇒ It is a recovering algorithm
- ⇒ Algorithm for Recovery and Isolation Exploiting Semantics
- ⇒ It is based on Write Ahead Log protocol
- ⇒ ARIES has 3 phases in order

1] Analysis

2] Redo

3] Undo

4 Types of Log record

<START>

<UPDATE>

<COMMIT>

<ABORT>

- ⇒ Log is most commonly used structure for recording the modification that is to be

log file is maintained

$\langle T, \text{start} \rangle$

$\langle T, a=10, 20 \rangle$

$\langle T, \text{COMMIT} \rangle$

Redo and Undo Operation

* There is an operation in which restore all the old value on to the disk called Undo operation or Rollback

* There is an operation in which all the modified values are restored to the disk, This is called Rollforward or Redo operation

Write Ahead Logging Rule (WAL)

⇒ Before a block of data in a main memory is a output to the database.

All log record ^{preexisting} in the block must have be output to the ^{stable} storage.

This rule is called WAL rule

Deferred Database Modification

No Redo / Undo Algorithm

T_1	T_2
Read(A, a) $a = a - 10$ // 90 Write(A, a) Read(B, b) $b = b + 10$ // 210 Write(B, b)	Read(c, c) $c = c - 20$ // 280 Write(c, c)

$A = 100$
 $B = 200$
 $C = 300$

$\rightarrow T_1$
 $\rightarrow T_2$

Immediate Db Modification

In this technique the database is updated during the execution of a transaction even before it reaches the commit point.

Log	Database
$\langle T_1, \text{start} \rangle$	
$\langle T_1, A \ 100 \ 90 \rangle$	$A = 90$
$\langle T_1, B \ 200 \ 210 \rangle$	$B = 210$
$\langle T_1, \text{COMMIT} \rangle$	
$\langle T_2, \text{START} \rangle$	
$\langle T_2, C \ 300 \ 280 \rangle$	$C = 280$
$\langle T_2, \text{COMMIT} \rangle$	

Unit - 4

Implementation Techniques

Primary Memory - Cache and Main memory
(Non Volatile) ↑ Faster than others (Expensive)

Secondary Memory - Magnetic Disk, Flash Memory

Tertiary Memory - Optical Disk, Magnetic Tape
(Data Backup)

Features of a Memory

* Accessing Speed

* Performance per unit of data

* Reliability

According to features storage is classified into different categories.

* Cache Memory

* Optical Disk

* Flash Memory

* Main Memory

* Magnetic memory

* Magnetic Tape

RAID (Redundant Array of Independent Disks)

→ It is a disk organization technique which manage a large no. of disks, providing a view of a single disk of

- 1] High capacity and High speed by using multiple disks in parallel
- 2] High reliability by storing data redundantly so that data can be recovered even if a disk fails.

⇒ RAID employs the technique of disk mirroring or disk striping which involves partitioning each drive's storage space into units ranging from a sector (512 bytes) into several megabytes.

Organization of Records in Files

- * It stores an organized form
- * Records must be in organized form

5 types

- i] Sequential File Organization
- ii] Heap File Organization
- iii] Hash File Organization
- iv] B+ Tree File Organization
- v] Clustered File Organization

1] Sequential File Organization

⇒ records are arranged in sequential order

Method's Type

1. Pile File Method

2. Sorted File Method

Adv

* Simple Design

* Cheaper

Disadv

* Fetching

Storage Mechanism

Particular Data

2] Heap File

Using some strategies in heap file

Adv

* Fetching of Record is easy

3] Hash File

function is

Hash performed over the records

4] B+ Tree

Tree structure to store data

5] Clustered File

Data Dictionary Storage

Data dictionary storage is mini database management system that manages met

Metadata

Metadata is a database information but not show the content

Roll No	Reg No	Name	Phone No.
2208001	7236221	XXX	9050184841
2208002	7236222	YYY	9750427203
2208003	7236223	ZZZ	9872654310

Ex Data Dictionary

Attribute	Datatype	Information
Roll No	int	Primary key of the table
Reg No	int	Reg No of the student
Name	varchar(50)	Name of the student
Phone No	int	Phone No of the student

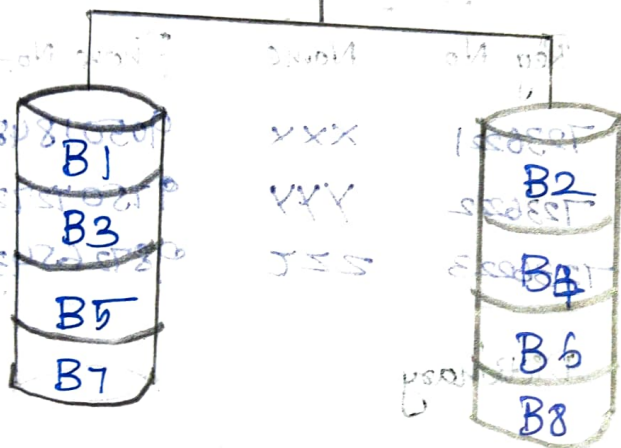
RAID LEVELS

RAID

- * Uses block level striping
- * Data loss is not critical

RAID 0

RAID 0



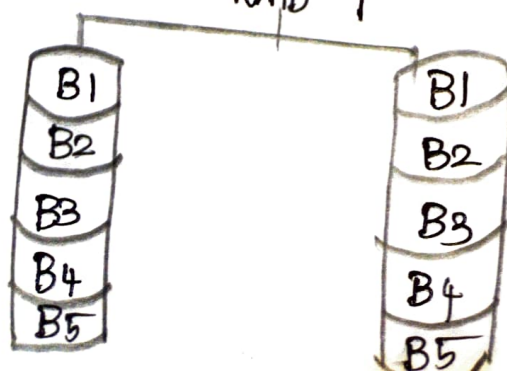
RAID

- * Uses disk mirroring

* Read performance is improved

Since either disk can be read at the same time

RAID 1

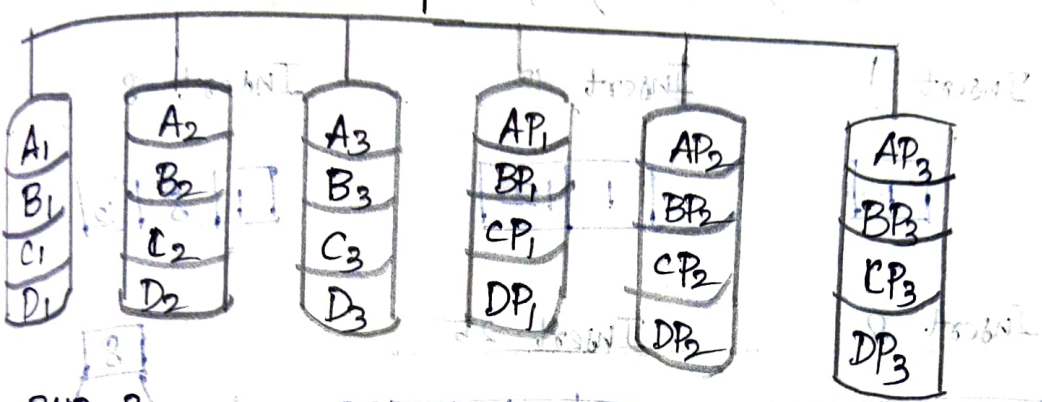


RAID 2

⇒ Rarely used in practice

⇒ High transfer rates are possible

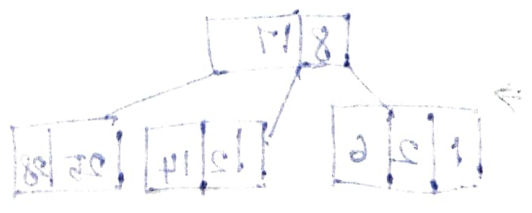
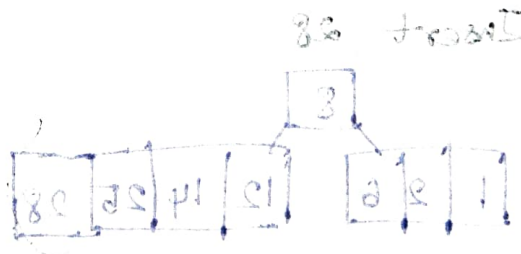
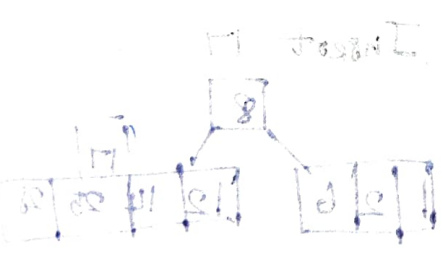
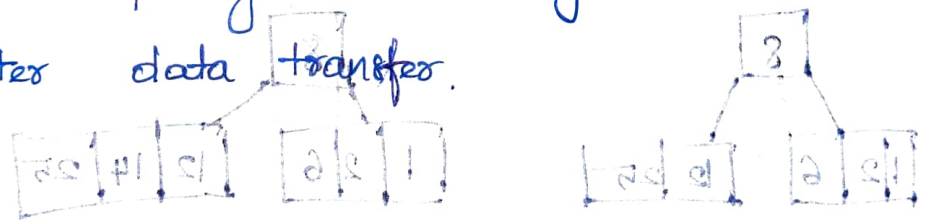
RAID 2



RAID 3

Consists byte level striping with a dedicated parity disk

Faster data transfer



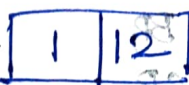
B Tree Index

1, 12, 8, 2, 25, 6, 14, 28, 17, 7, 52, 16, 48, 68,
3, 26, 29, 53, 55, 45, 67

Insert 1



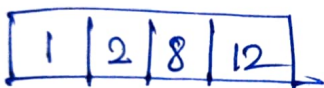
Insert 12



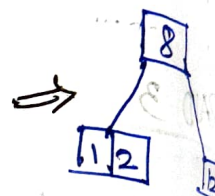
Insert 8



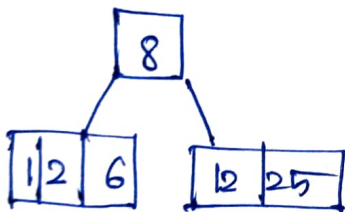
Insert 2



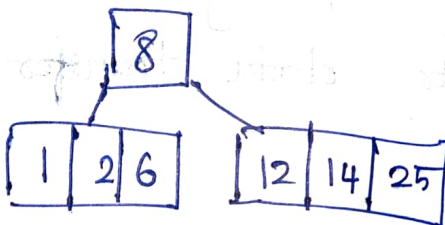
Insert 25



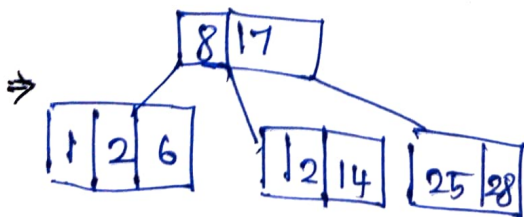
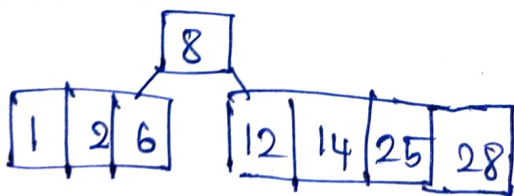
Insert 6



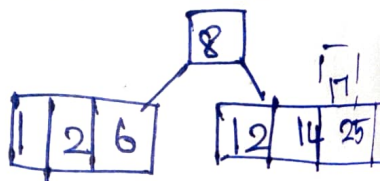
Insert 14



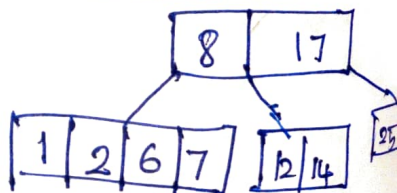
Insert 28



Insert 17



Insert 7



B Tree Index File

1] Select Operation

81 Linear Search (Brute Force)

$$\text{Cost} = \frac{br}{2} + 1 \text{ Seek}$$

cost for

82 Primary Index

$$\text{Cost} = (h_i + 1) * (\#T + \#S)$$

83 Non-key Attribute

$$\text{cost} = h_i(\#T + \#S) + \#S + \#T * b$$

84 Secondary Index

$$\text{Cost} = (h_i + 1) * (\#T + \#S)$$

2] Join Operation

Types

1] J_1 Nested Loop Join

2] J_2 Single Loop Join

3] J_3 Sort - Merge Join

4] J_4 Hash-Join

Primary Index

In primary index, there is one relationship between the entries in the index table and the records in the main table.

Dense Primary Index

* The no. of entries in the index is the same as the no. of entries in the main table.

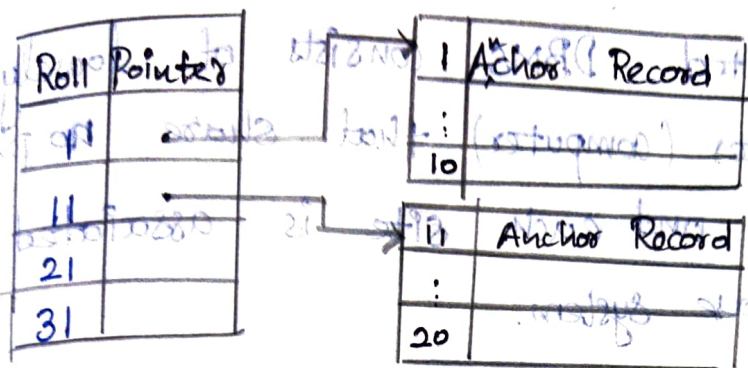
* In other words, each and every record in the main table has an entry in the index.

Roll	Pointer	Roll
1	→	1
2	→	2
3	→	3
4	→	4

E sparse (or) Non-dense Primary Index

For larger tables, the dense primary index itself begins to grow in size. To keep the size of the index smaller,

Instead of pointing to each and every record in the main table, the index points to the records in the main table in a gap.



Single Level Indexing

* A single value index is an auxiliary file that makes it more efficient to search for a record in the datafile.

Multi level Indexing

There is an even immense need to keep the index record in the main memory so as to speed up the operations. If a single level index is used then a large size index cannot be kept in the memory which leads to multiple disk access.

Secondary Index

In this technique two levels of indexing is used in order to reduce the mapping size of the first level and in general

Unit - 5

ADVANCED TOPICS

Distributed Databases

A distributed DBMS consists of loosely coupled sites (computer) that share no physical components and each site is associated a database system.

Object Relational and NoSQL Database -5

Mapping EER to ODB schema :-

Step 1: Create an ODL class for each EER entity type or subclass

Step 2: Add relationship Properties (or) reference attributes for each binary relationship

Step 3: Include appropriate operation for each class

Object Identifier :-

In a database OID is a set of Integers that uniquely identifies each Row in a table. OID is typically is a sequence of numbers separated by period



Properties of OID :-

- * It is not visible to the external user
- * It is Immutable
- * OID value of a particular object should not change
- * Even if an object is removed from the database the OID should be preserved

Reference Types :-

* The "Ref is System Generated" statement indicates that whenever a new person-type object is created, the system will assign it a unique system-generated identifier.

* But it is also possible not have a system generated object identifier and use the traditional key of the basic relational model is desired.

Object Identifier using Reference types :-

* The user can specify the system-generated object identifier for the individual row by using

Ref is $\langle \text{old Attribute} \rangle \langle \text{Value-Generation-method} \rangle$;

* The option for $\langle \text{value-generation-method} \rangle$

System generated : in which case the system will automatically generate a unique identifier

Derived : in which the traditional method of

using the user-provided Primary key value to identify tuples is applied

Supertype and subtype:

Supertype:- is an entity type that has got relationship with one (or) more subtypes and it contains attributes that are common to its subtypes.

Subtype are subgroups of the supertype entity and have unique attributes but they will be different from each subtype.

Eg: People, Bank Account, Asset, Liability, Credit Card.

User defined Routines:

The system creates the framework a form routine for you with the name Prefix `ur`. This routine is reusable, which means that you can use in other object in the Project.

Rowtype:-

Rowtype attribute provides a record type that represents a row in a database table. The record can store an entire row of data selected from the table (or) fetched from a cursor (or) cursor variable field in a record and corresponding columns in a row have the same names and the datatypes.

UDT:-

* User defined data type (UDT) is a data type that derived from an existing data types. You can use UDTs to extend the built-in types already available and create your own customized data types.

- Distinct type
- Structured type
- Reference type
- Array type
- Row type

Distinct type:-

* A distinct type is a UDT that shares a internal representation with an existing built in types

PL/SQL collection :-

* A collection is an ordered group of element having the same data type. Each element is identified by unique subscript that represent its position in the collection.

- * Nested table
- * Index by table
- * Variable size array.

Index by table :-

Index by table is a set of key value pairs

Nested table :-

* nested tables is like one-dimensional array with an arbitrary number of elements

Collection method :-

PL/SQL provides the built in collection method that make collection easier to use.

Object query language :-

OQL is a version of the SQL that has been

designed for use in Network managers. The components create and interact with their database using OQL. use OQL to create new database (OR) insert data into existing database.

Convention and sample database:-

* To illustrate the OQL keywords in use; a sample database has been used, the staff database.

Features of OQL

- * Database and table creation
- * Inserting data into a table
- * Selecting data from a table
- * Counting rows in a table

NoSQL:-

* NoSQL databases store data differently than relational tables

* NoSQL databases come in a variety of types based on their data model.

Types of database - NoSQL

Document oriented database :-

* It stores data in documents similar to:

JSON.

* Each document contains Pair of field and

values.

* The values can typically variety of other

Object

Key value database :-

* Key value store is a simpler type of database where each item contains key and

values

* Each Key is unique and associated with a single value

CAP Theorem :-

* CAP Theorem is fundamental concept in distributed system theory.

Consistency :-

* It means all the nodes inside a network will have the same copies of a replicated data item visible for various transactions

Availability :-

* It means that each read (or) write request for a data item will either be processed successfully (or) will receive a message that the operation cannot be completed.

Partition Tolerance :-

Partition tolerance means that the system can continue operating even if the network connecting the nodes has a fault that results in two or more partitions where the nodes in each partition can only communicate among each other.

Document based : mongo DB. Data model and

CRUD Operations :-

(Create, Read, update and Delete) are the basic set of operations that allow users to interact with the mongoDB server.

Perform CRUD operation in MongoDB.

1. Create operation :-

* The create (or) insert operation used to insert (or) add new document in the collection

db. create collection () It is used to create an empty collection

Column-based Hbase data model and CRUD operation :-

* Hbase is able to do what an RDBMS cannot host very large, sparsely populated tables on cluster made from commodity hardware.

* HBase is a column-oriented database

~~* The~~
Shared-nothing Architecture :-

* When building Scalable system, we need to parallelize our computations

* This requires multiple processors

Sharding :-

* HBase Provides automatic Sharding, with minimal duplication of state

Regions :-

* Tables are Partitioned vertically into different regions

The Hadoop Distributed file system :-

* In a production configurations HBase stores its underlying data on a file system called HDFS

* HDFS uses a cluster of computers of simulate a single

→ Resilient against hardware

→ Streaming Data access

→ Support for a large file