Introduction to RDBMS and SQL Concepts

RDBMS Concepts

1 Introduction to RDBMS

2 Entity - Relationship Diagrams

3 Schema Refinement and Normalization

4 Introduction to Oracle SQL

5 Scoping and Ordering of Rows

6 DML Statements

7 Group Functions, Group by, & Joins

Introduction to RDBMS

Module 1

Objectives

At the end of this module, you will be able to:

- Define Database Management System (DBMS)
- Identify benefits of database approach
- Recognize functionalities of Database System
- Differentiate Data Models
- Explain three level DB Architecture
- Define & Identify features of an RDBMS
- Identify features of CODD's Principles / Rules
- Define terms like Table, Tuple, Attribute, Primary Key …
- Describe properties of Relations, Keys and Referential Integrity

Duration: 2 hrs

DBMS: Definitions

- Database Management System is one of the oldest technique, which contains a set of programs specially designed for creation and managing of DATA stored in a DATABASE
- Data
  - A known fact that can be recorded and that have implicit meaning
- Database
  - A collection of related data with the following implicit properties
    - A Database is a logically coherent collection of data with some inherent meaning
    - A Database is designed, built, and populated with data for a particular purpose
- Database System
  - Database and DBMS software integrated to work together forms a database system

DBMS

- Goals of a Database Management System
  - To provide an environment which will efficiently provide access to DATA in Database
  - Implement Security, Control in Concurrency and Recovery from Crash
- It is a general purpose facility for:
  - To define Database
  - To construct Database
  - Manipulate Database

Benefits of Database Approach

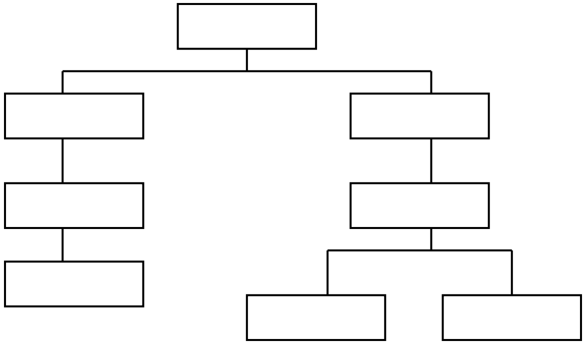
- Easy access of DATA
- Redundancy is reduced
- Inconsistency is avoided
- Data is shared
- Standard's are enforced
- Security is applied
- Integrity is maintained
- Data independency is provided
- Abstract view of DATA
- Multiple and simultaneous access of DATA

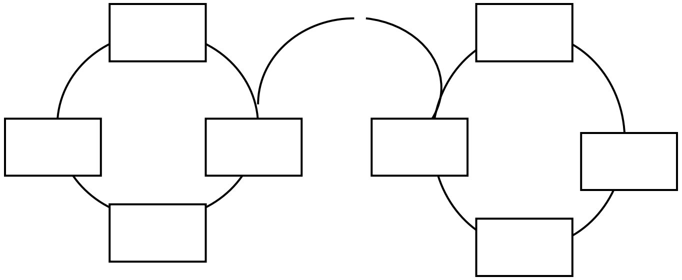
History of Database / Evolution of Database

Mid 1940’s	-	File System
Mid 1950’s	-	Hierarchical Database System
Mid 1960’s	-	Network Database System
Mid 1970’s	-	Relational Database System (Mathematical Set Theory)
Mid 1980’s	-	Object Database System
Mid 1990’s	-	Object Relational Database System

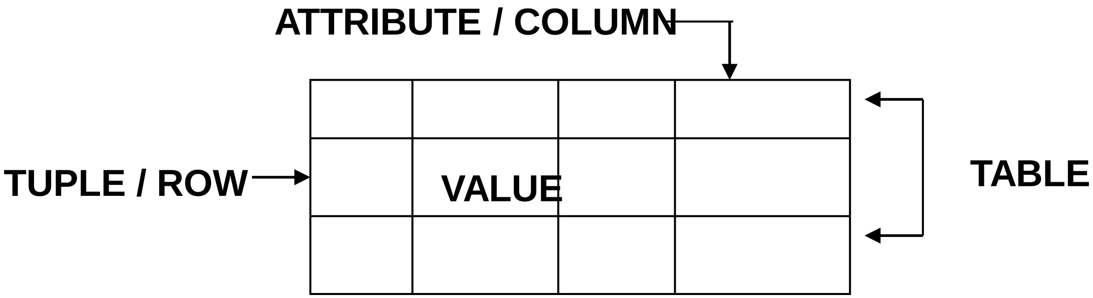
Types of Database Models

HIERARCHICAL



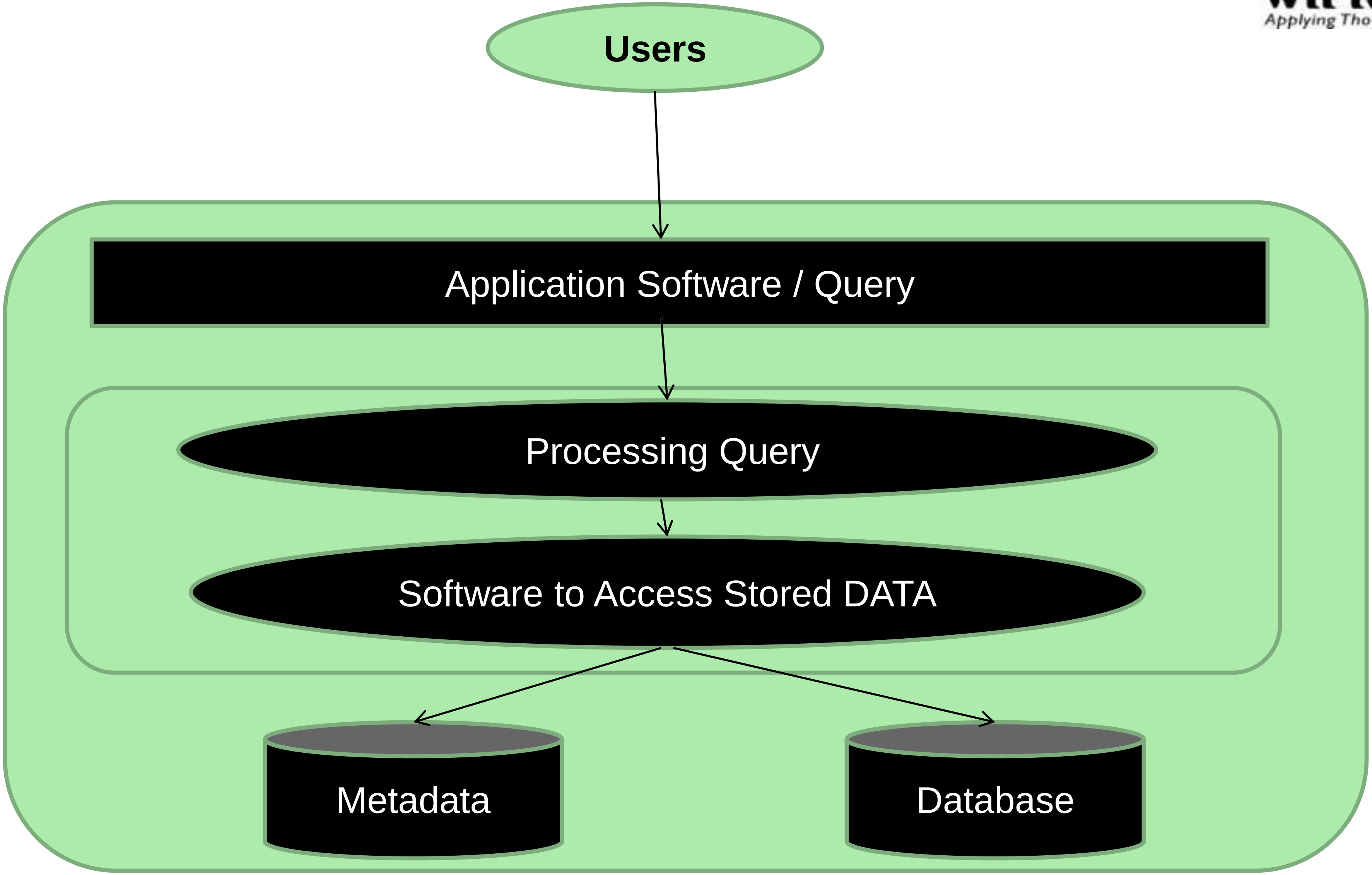


NETWORK

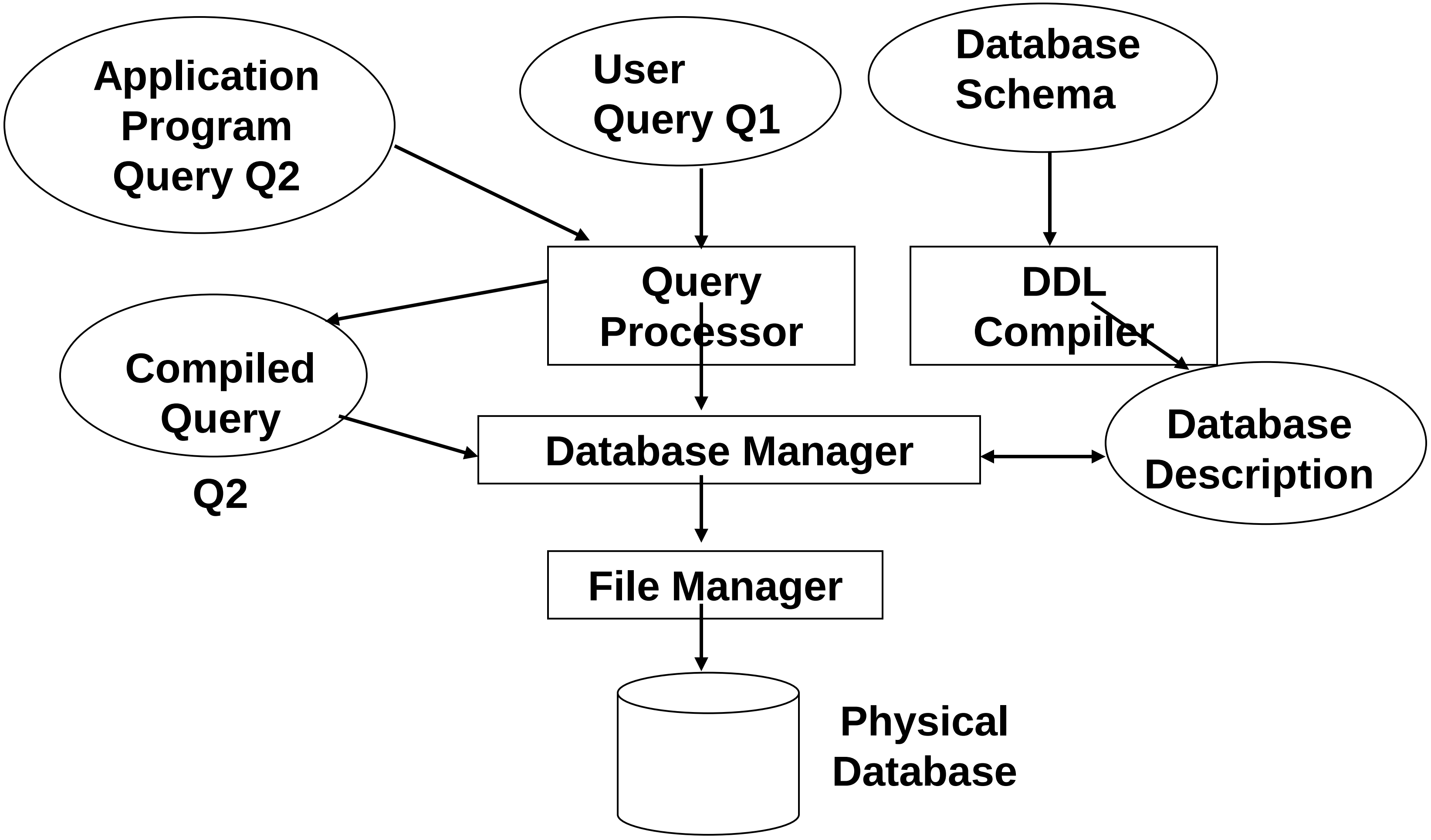


RELATIONAL

Database System



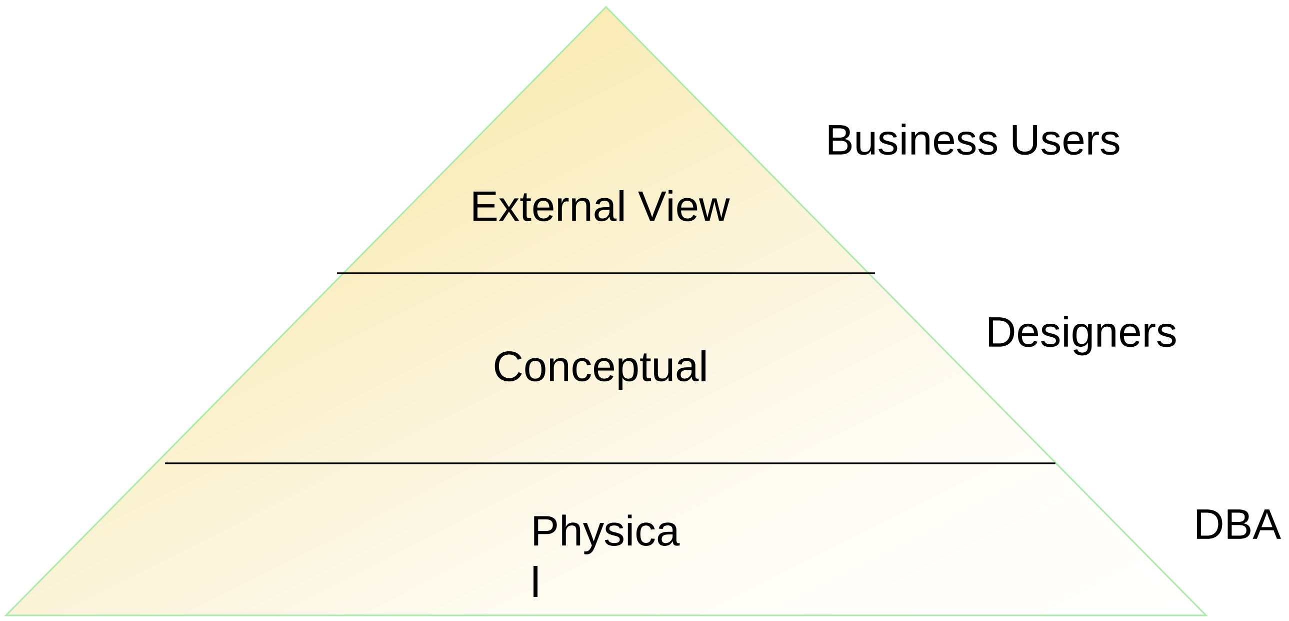
Database System (Contd.).



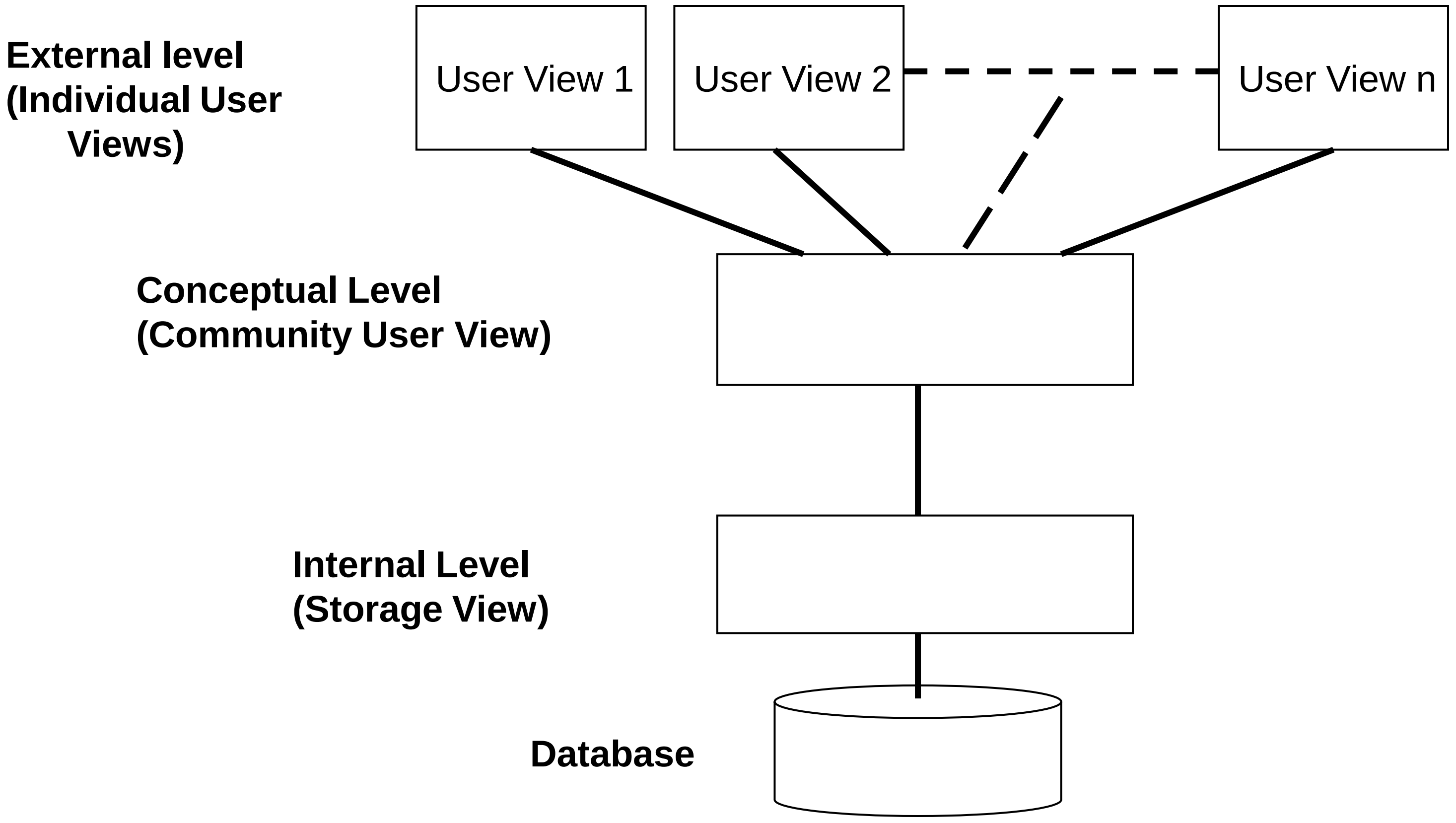
Database System Model

Database System provides three DATA models

- External View
- Conceptual
- Physical



Database Architecture Levels



An Example of three Levels

CONCEPTUAL VIEW

ENO	NAME	F-NAME	AGE	SALARY	DEPTNO

EXTERNAL VIEW

ENO	NAME	F-NAME	AGE

ENO	SALARY	DEPTNO

```
struct Employee
{
int eno;
char name(20);
char f_name(20);
float salary;
int deptno
struct Employee *ptr
};
```

Internal View

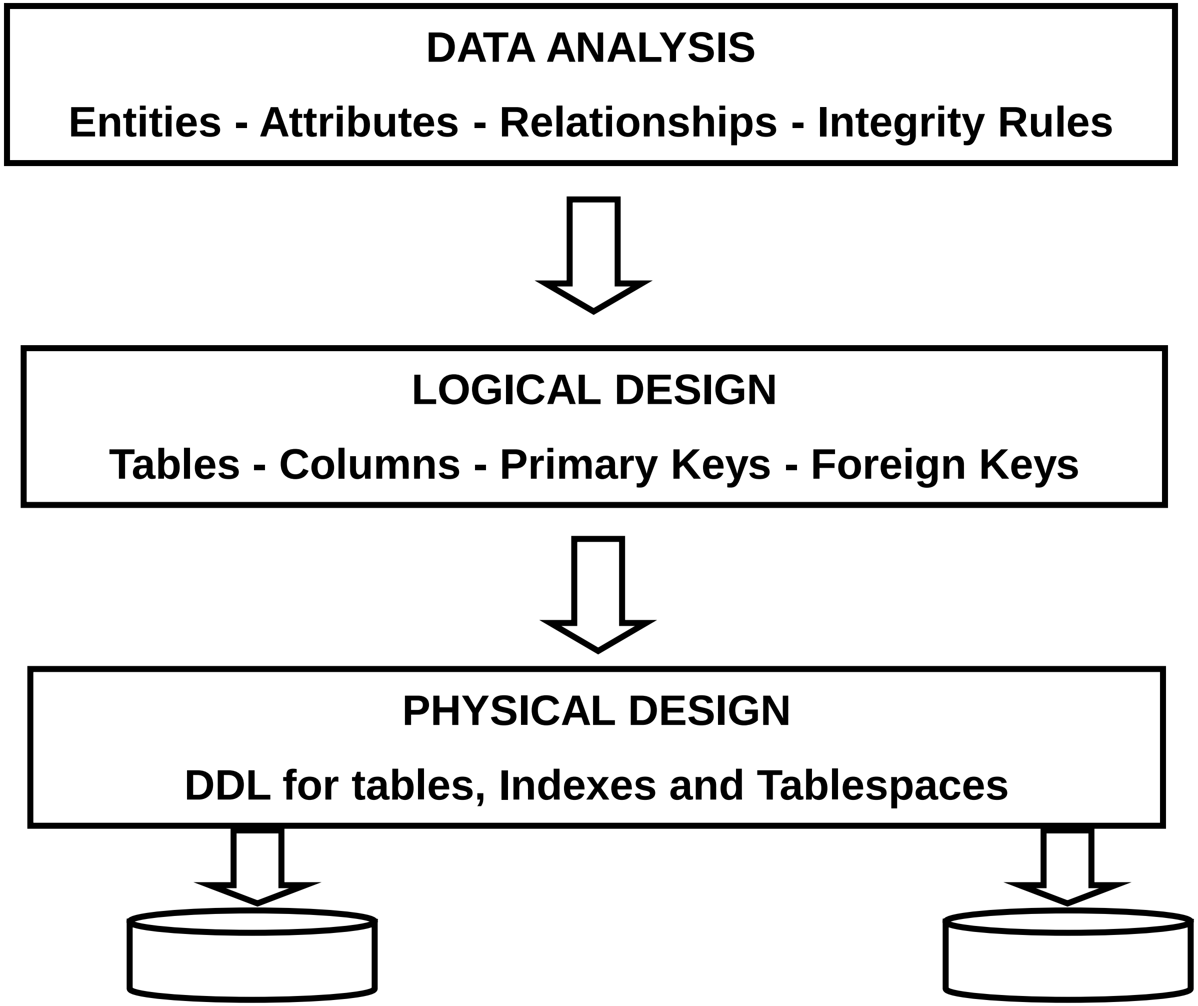
Schema

- **Schema**: Description of DATA in terms of a DATA Model
- Three-Level DB Architecture defines following schemas:
  - **External Schema (or sub-schema)**
    - Writing using external DDL
  - **Conceptual schema**
    - Writing using conceptual DDL
  - **Internal Schema**
    - Writing using internal DDL or Storage Structure Definition

Data Independence

- Change the schema at one level of a Database System without a need to change the schema at the next higher level
  - **Logical DATA Independence**: Refers to the immunity of the external schemas to changes in the conceptual schema E.g., Add new record or field
  - **Physical DATA Independence**: Refers to the immunity of the conceptual schema to changes in the internal schema E.g., Addition of an index should not affect existing one

Database Design Phases



DATA ANALYSIS

Entities - Attributes - Relationships - Integrity Rules

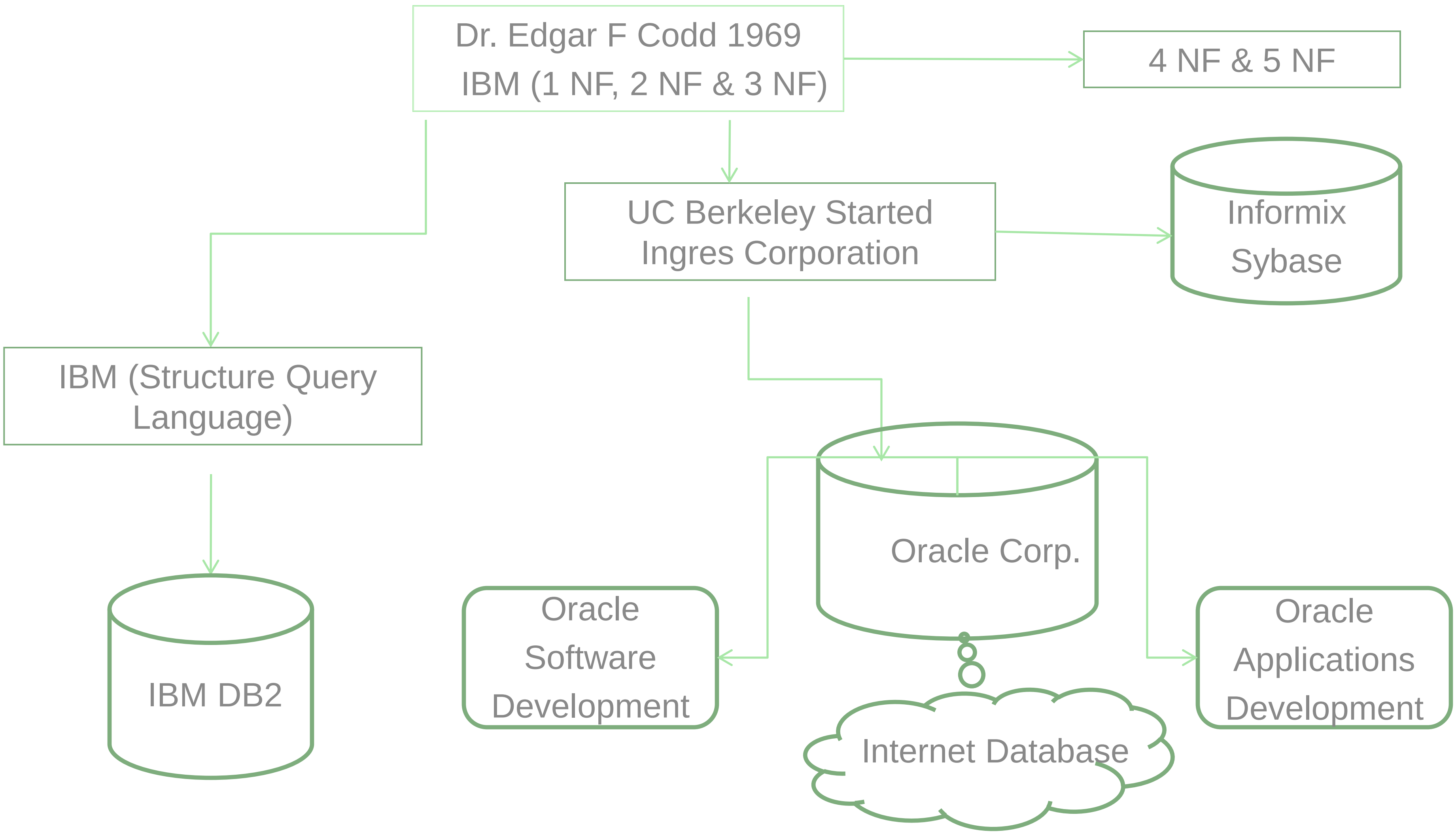
LOGICAL DESIGN

Tables - Columns - Primary Keys - Foreign Keys

PHYSICAL DESIGN

DDL for tables, Indexes and Tablespaces

History of Relational Database System



Dr. Edgar F Codd 1969
IBM (1 NF, 2 NF & 3 NF)

4 NF & 5 NF

UC Berkeley Started Ingres Corporation

Informix
Sybase

IBM (Structure Query Language)

IBM DB2

Oracle Corp.

Oracle Software Development

Oracle Applications Development

Internet Database

Definition of RDBMS

- RDBMS is a Relational Data Base Management System or simply Relational DBMS introduced by E.F. Codd which adds an additional condition that the system supports a tabular structure for the data with enforced relationships among the tables

- RDBMS
  - It is a system in which at a minimum:
    - Data accessed by the user with use of relation (Availability of DATA in tabular form, i.e. as a collection of tables, where each table consisting a set of rows and columns)
    - Provides a SET of relational operator to manipulate the DATA in tabular form.
  - Features of an RDBMS
    - The ability to create multiple relations (tables) and enter data into them.
    - An interactive query language.
    - Retrieval of information stored in more than one table.

Features of an RDBMS

- The ability to create multiple relations (tables) and enter data into them
- An interactive Query Language
- Retrieval of information stored in more than one table
- Provides a Catalog or Dictionary, which itself consists of tables (called system tables)

CODD’s Principles / Rules

0. The Rule specifies the system must qualify as relational and use its relational facilities to manage / access database

1. All information in a relational database which includes table names, column names are represented by values in tables

2. Every quantity of data in a the relational database can be accessed by using combination of a table name, a primary key value which identifies the row and attribute name that is identified as a cell

3. The RDBMS provides a mechanism which handles records that have unknown or inapplicable values in a pre-defined fashion

CODD's Principles / Rules (Contd.).

4. The descriptions / information of a database and in its contents are database objects (tables) and therefore can be queried via the data manipulation language (DML)

5. A RDBMS System should allow user to do all of the following: Define tables and views, query and update the data, set integrity constraints, set authorizations and define transactions

6. RDBMS provides a mechanism to update any view which is theoretically updateable. Data consistency is ensured, since the changes made in the views are transmitted to the original table and vice-versa

7. The RDBMS provides mechanism for insertions, updation and deletion at a table level. The performance is improved since the commands act on a set of records rather than one record at a time

CODD's Principles / Rules (Contd.).

8. The execution of adhoc requests and application programs is not affected by any changes in the physical data access and their storage methods

9. Logical changes in any tables and views such adding / deleting attributes or changing fields lengths need not demand modifications in the programs or in the format of adhoc requests

10. Like database objects table / view definition, integrity constraints are stored in the on-line catalog and therefore can be changed without forced changes in the application programs

11. Application programs and adhoc requests are not affected by change in the distribution of physical data and structure

12. If the RDBMS provides a language that accesses the information of a record at a time, then this language should not be used to bypass any integrity constraints.

Some Important Terms

- **Relation :** A Table
- **Tuple :** A Row in a Table
- **Attribute :** A Column in a Table
- **Degree :** Number of Attributes
- **Cardinality** : Number of Tuples
- **Primary Key :** A unique identifier for the Table
- **Domain :** A pool of values from which specific attributes of specific relations draw their values

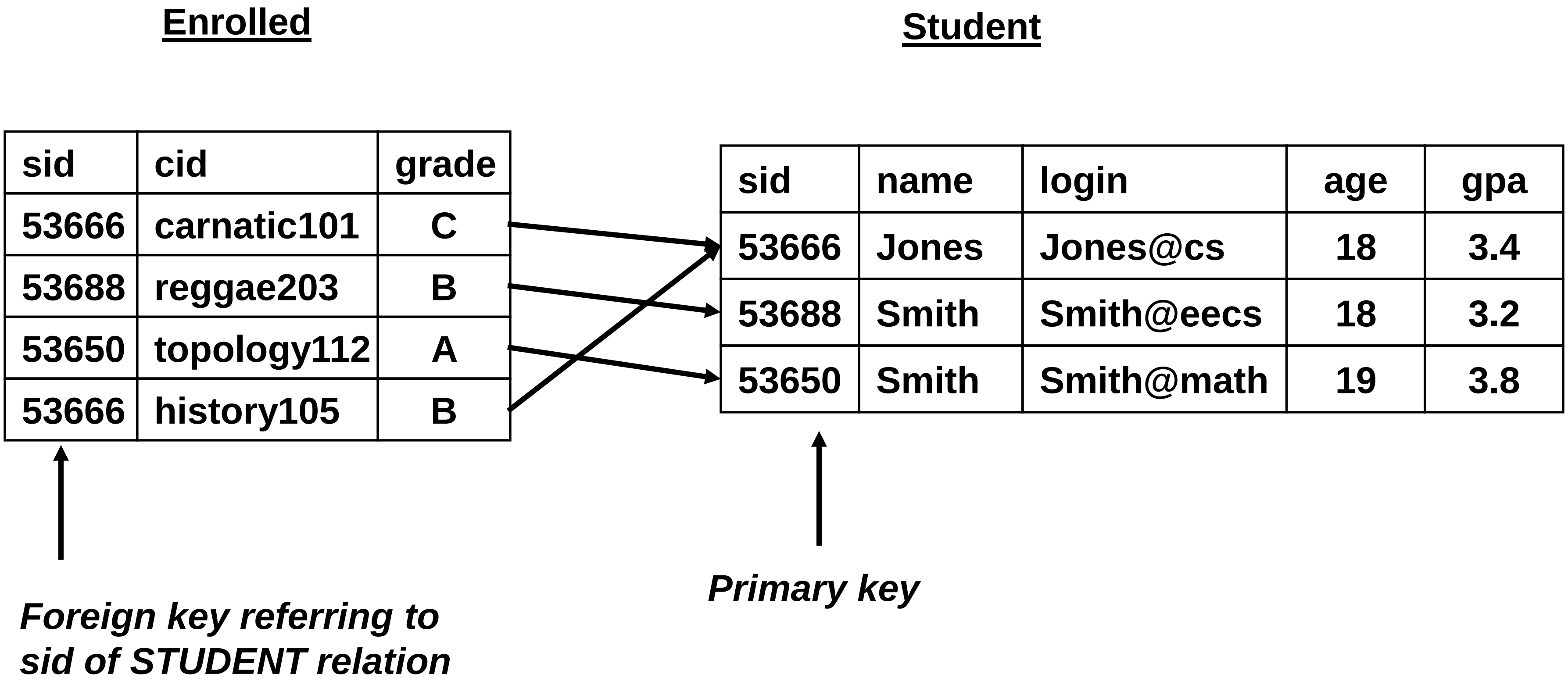
Relations or Tables properties

- There are no duplicate rows (Tuples)
- Tuples are unordered, top to bottom
- Attributes are unordered, left to right
- All attribute values are atomic (or scalar)
- Relational databases do not allow repeating groups

Keys

- Key
- Super Key
- Candidate Keys
  - Primary Key
  - Alternate Key
- Secondary Keys

Keys and Referential Integrity



Enrolled

sid	cid	grade
53666	carnatic101	C
53688	reggae203	B
53650	topology112	A
53666	history105	B

Student

sid	name	login	age	gpa
53666	Jones	Jones@cs	18	3.4
53688	Smith	Smith@eecs	18	3.2
53650	Smith	Smith@math	19	3.8

Foreign key referring to sid of STUDENT relation

Primary key

Points to Ponder

- Database Management System (DBMS)
- Benefits of database approach
- Functionalities of Database System
- Data Models
- DB Architecture
- Logical & physical data independence
- Features of an RDBMS
- CODD's Principles / Rules
- Table, Tuple, Attribute, Primary Key
- Properties of Relations
- Keys and Referential Integrity
- Constraints

Entity-Relationship Diagrams

Module 2

Objectives

At the end of this module, you will be able to:

- Explain E-R model and E-R designing phases
- Recognize ER modeling notations in terms of representing relations with constraints
- Model Entity Relationship diagram
- Draw Entity Relationship diagram

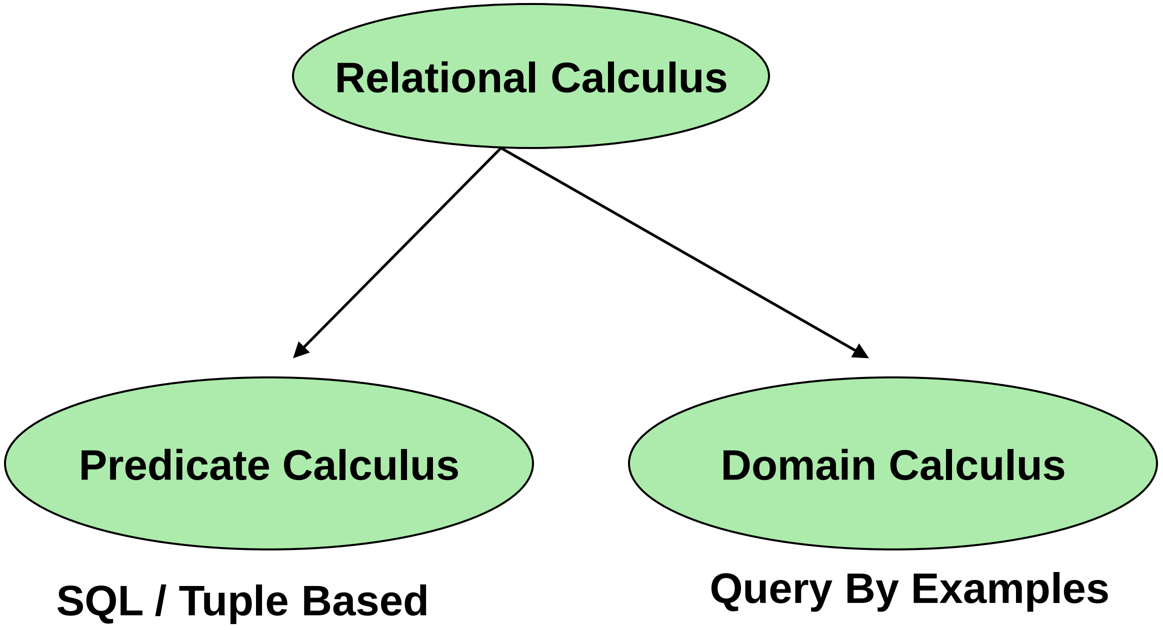
Duration: 3 hrs

Database Design: Overview

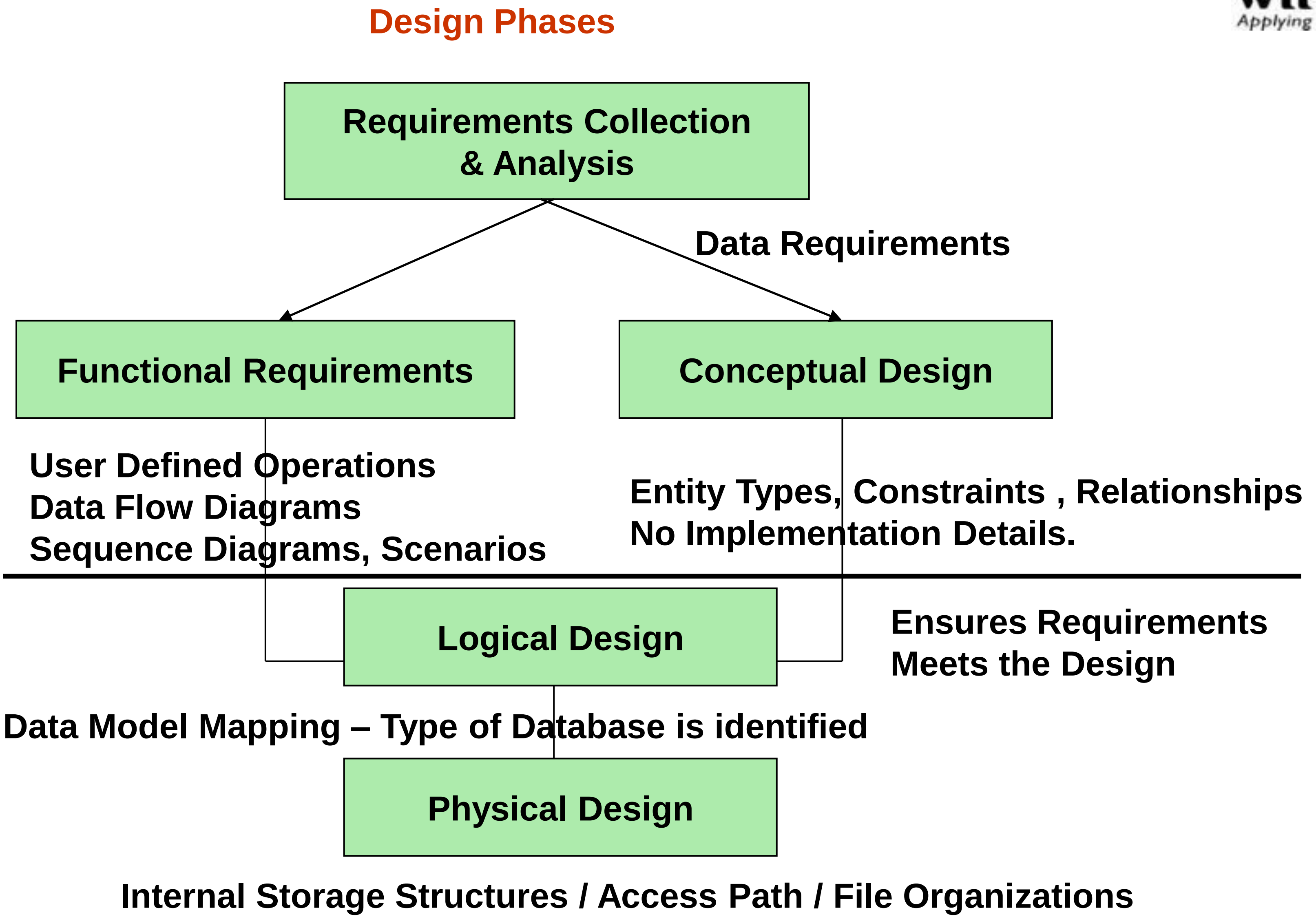
- Conceptual design : (Using ER Model)
- Fine-Tuning of Schema : (Normalization)
- Physically designing Database and Tuning

E- R Modeling

- Conceptual Schema Design
- Relational Calculus
  - Formal Language for Relational D/B.



E-R Modeling: Design Phases



Design Phases

Requirements Collection & Analysis

Data Requirements

Functional Requirements

Conceptual Design

User Defined Operations
Data Flow Diagrams
Sequence Diagrams, Scenarios

Entity Types, Constraints , Relationships
No Implementation Details.

Logical Design

Ensures Requirements
Meets the Design

Data Model Mapping – Type of Database is identified

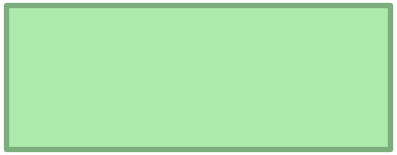
Physical Design

Internal Storage Structures / Access Path / File Organizations

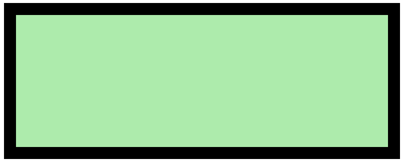
E-R Modeling: Terminology

- Entity
  - is anything that exists and is distinguishable
- Entity Set
  - a group of similar entities
- Attribute
  - properties that describe an entity
- Relationship
  - an association between entities

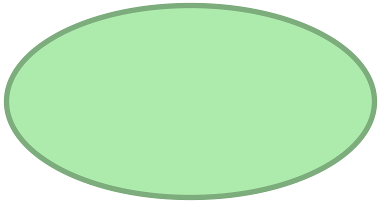
ER Modeling Notations



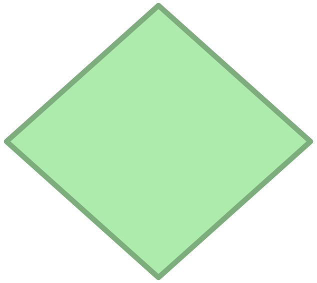
ENTITY TYPE



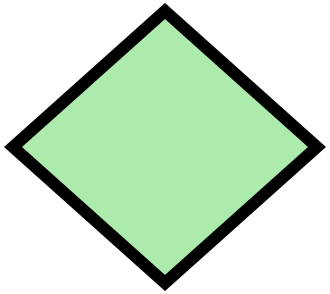
WEAK ENTITY TYPE



ATTRIBUTE TYPE

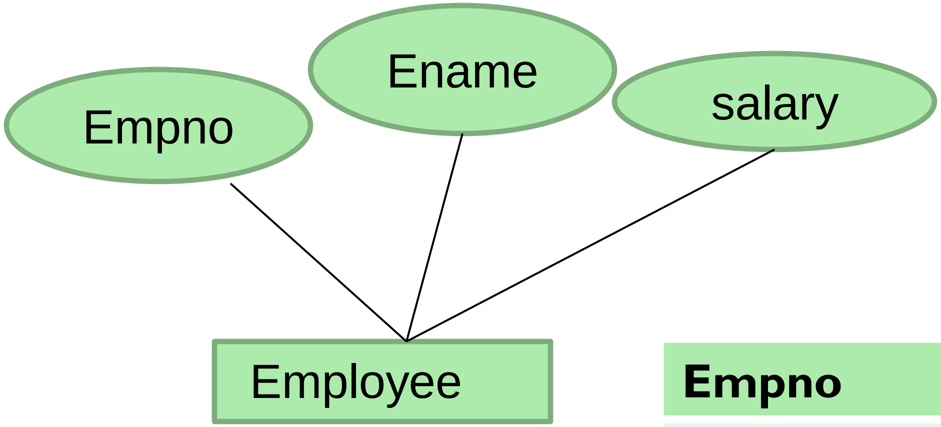


RELATIONSHIP TYPE



WEAK RELATIONSHIP TYPE

ER Modeling: Entity

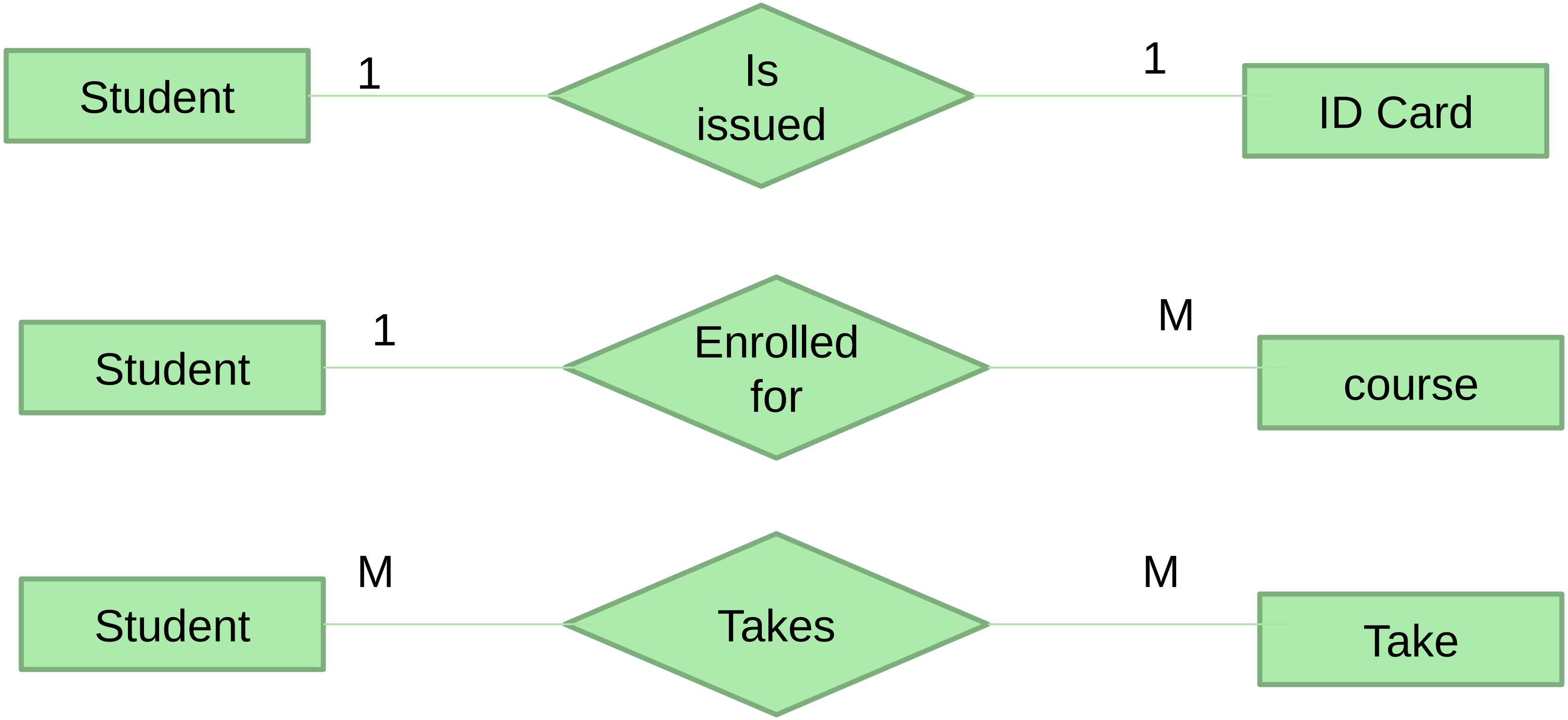


Entity set

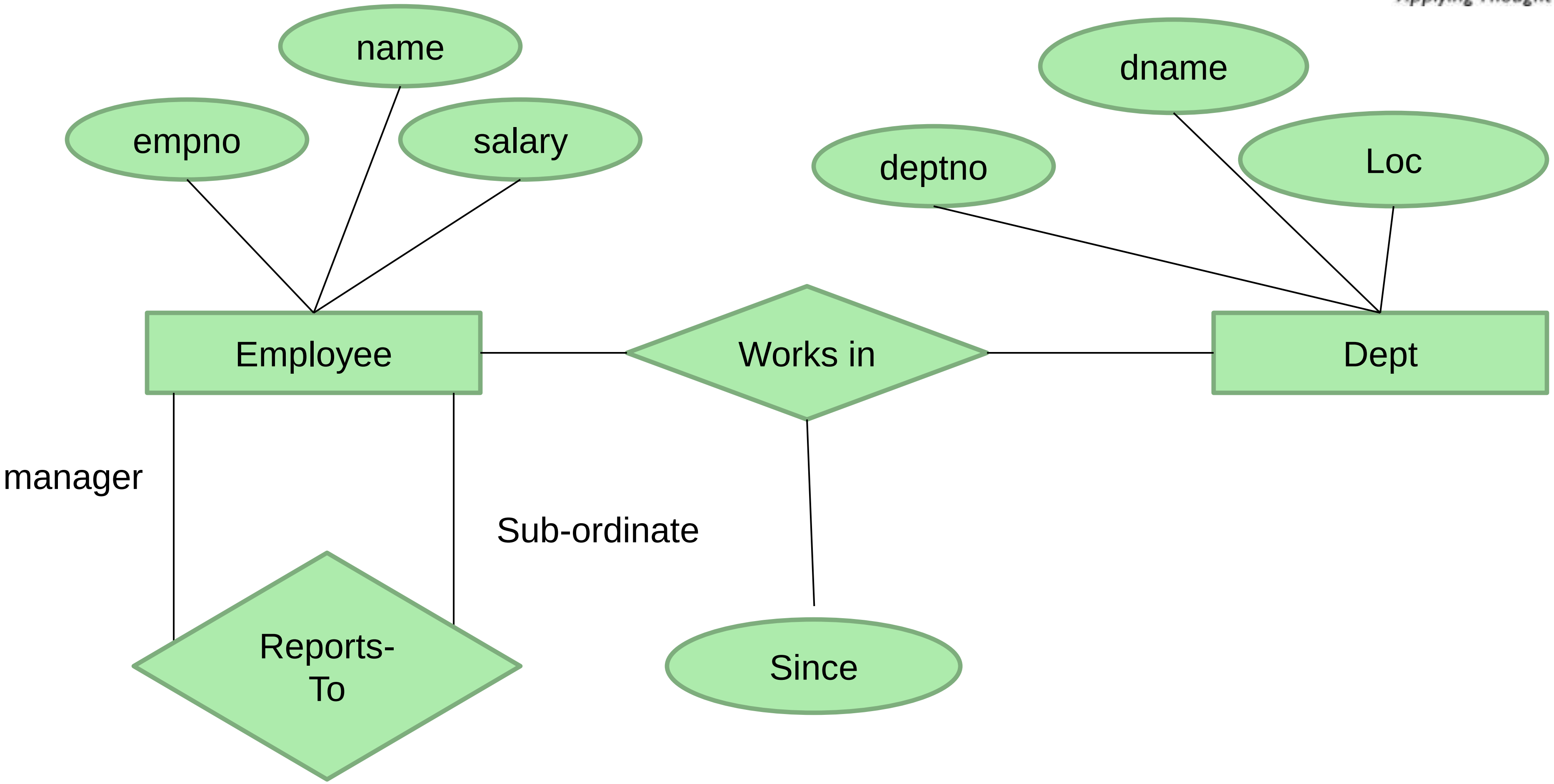
Employee
table

Empno	**Ename**	**salary**

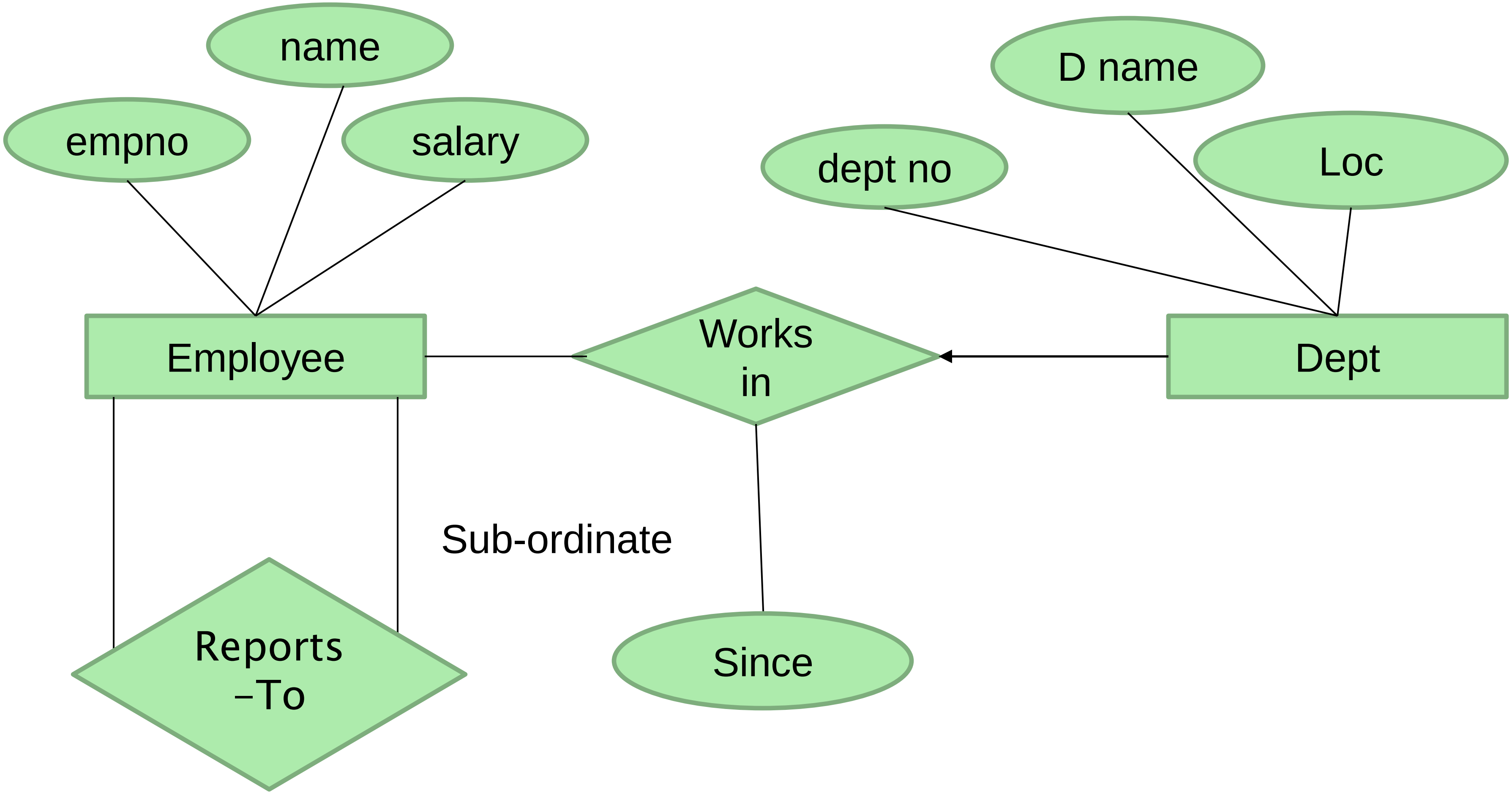
ER Modeling: Types of Relationships



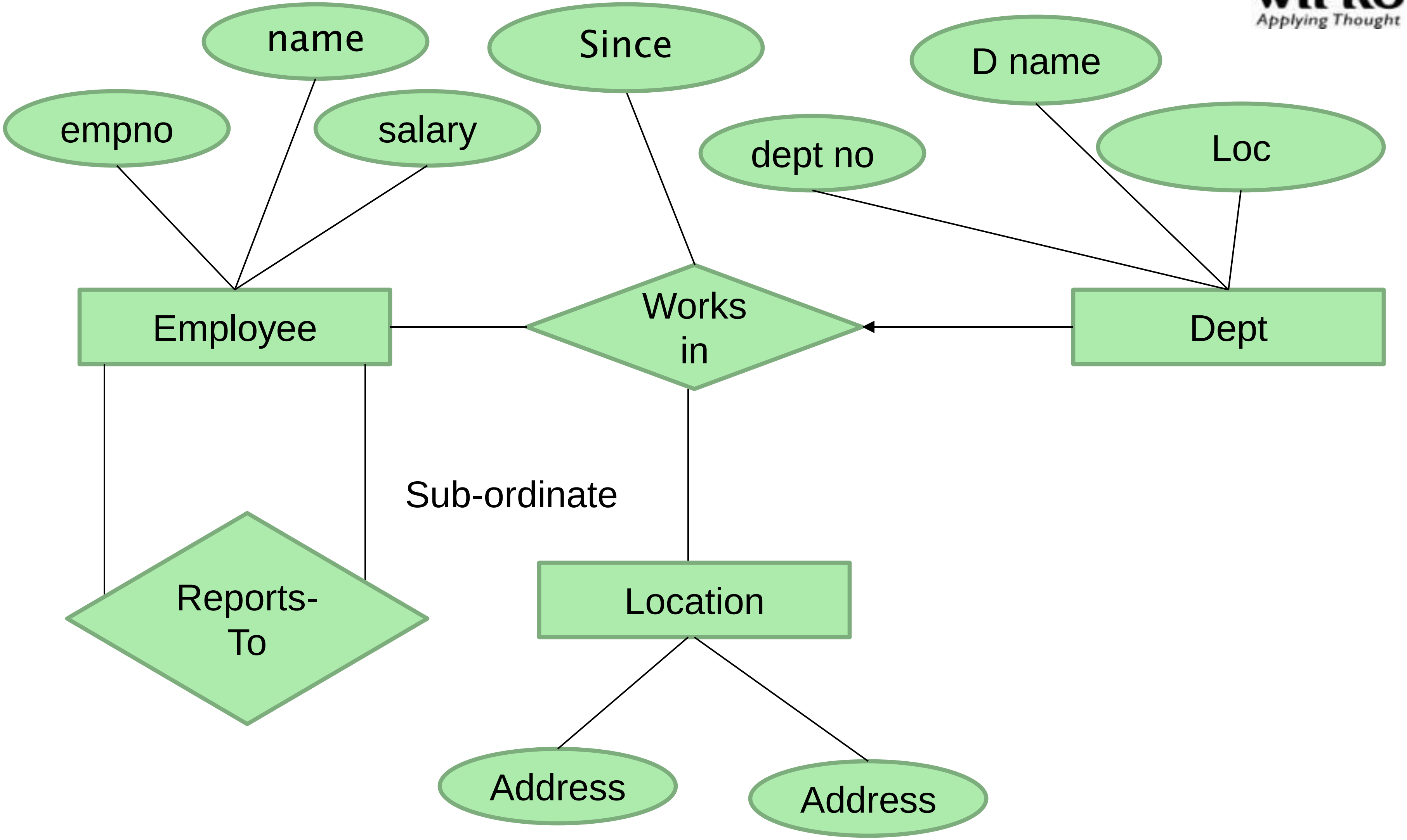
E-R Modeling E-R Model



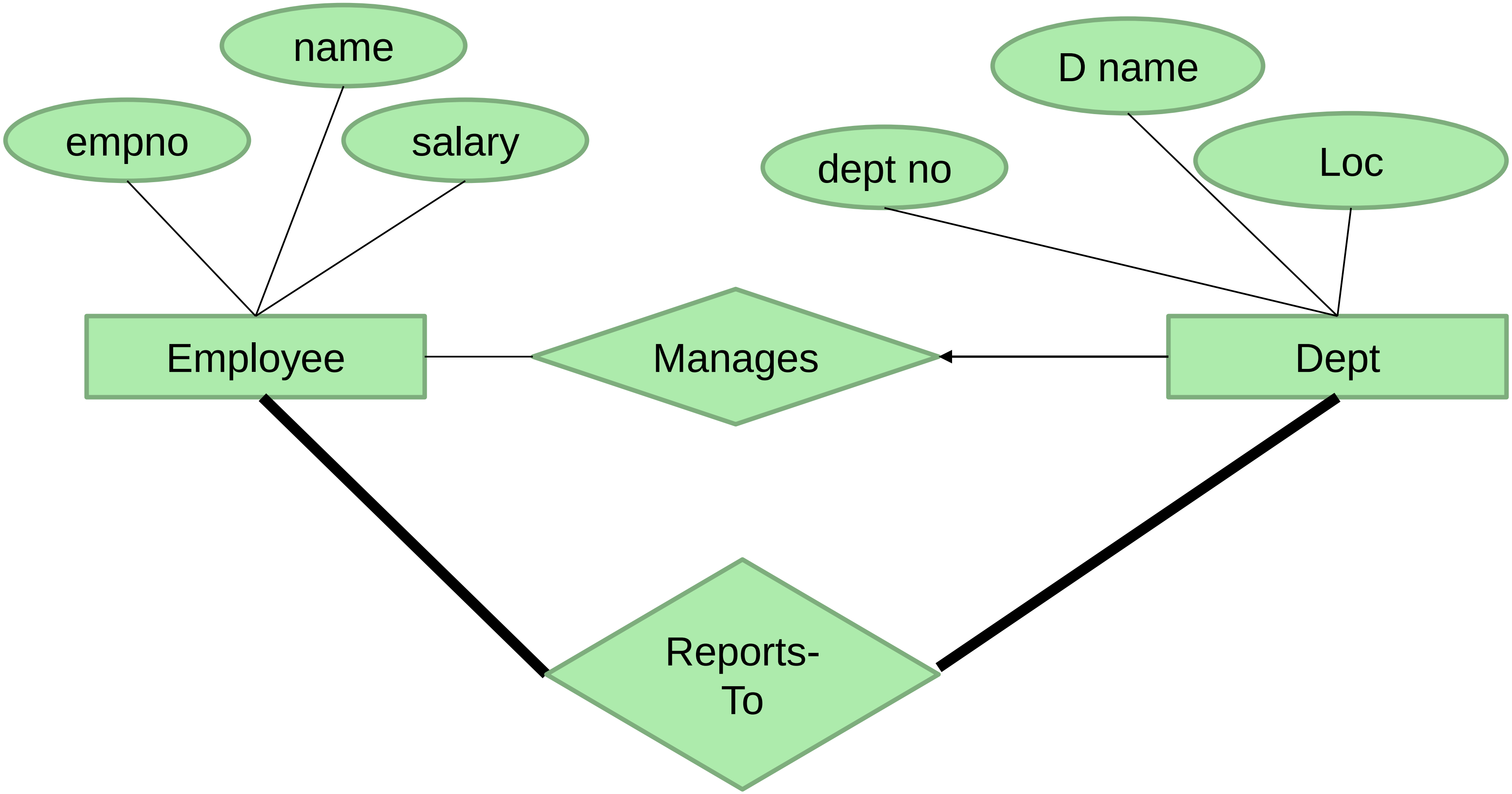
ER-modeling : Key constraint



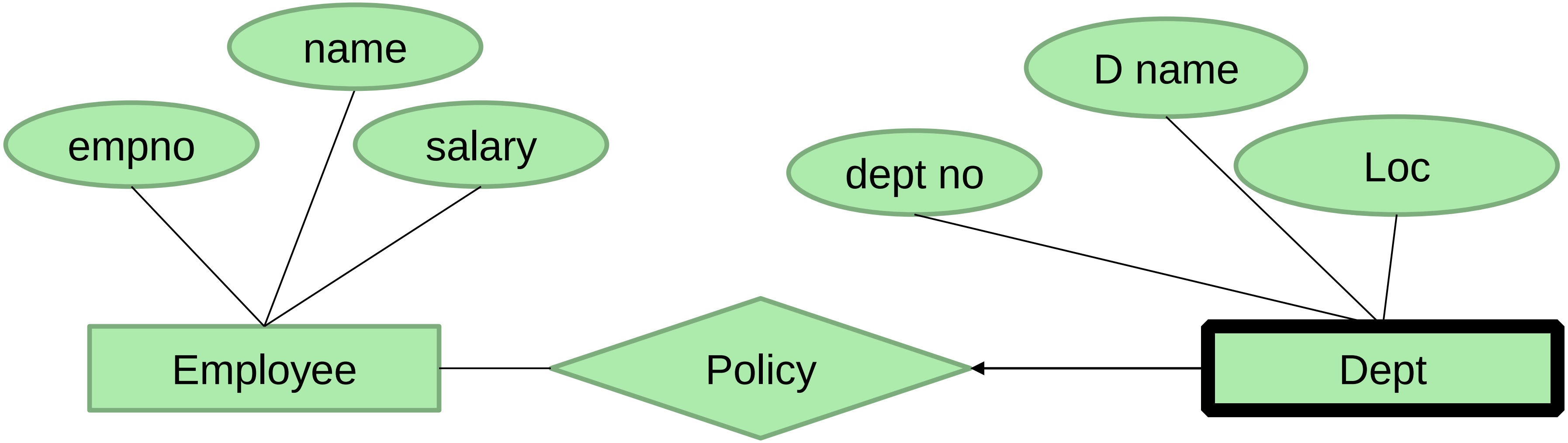
Key constraints for Ternary Relationships



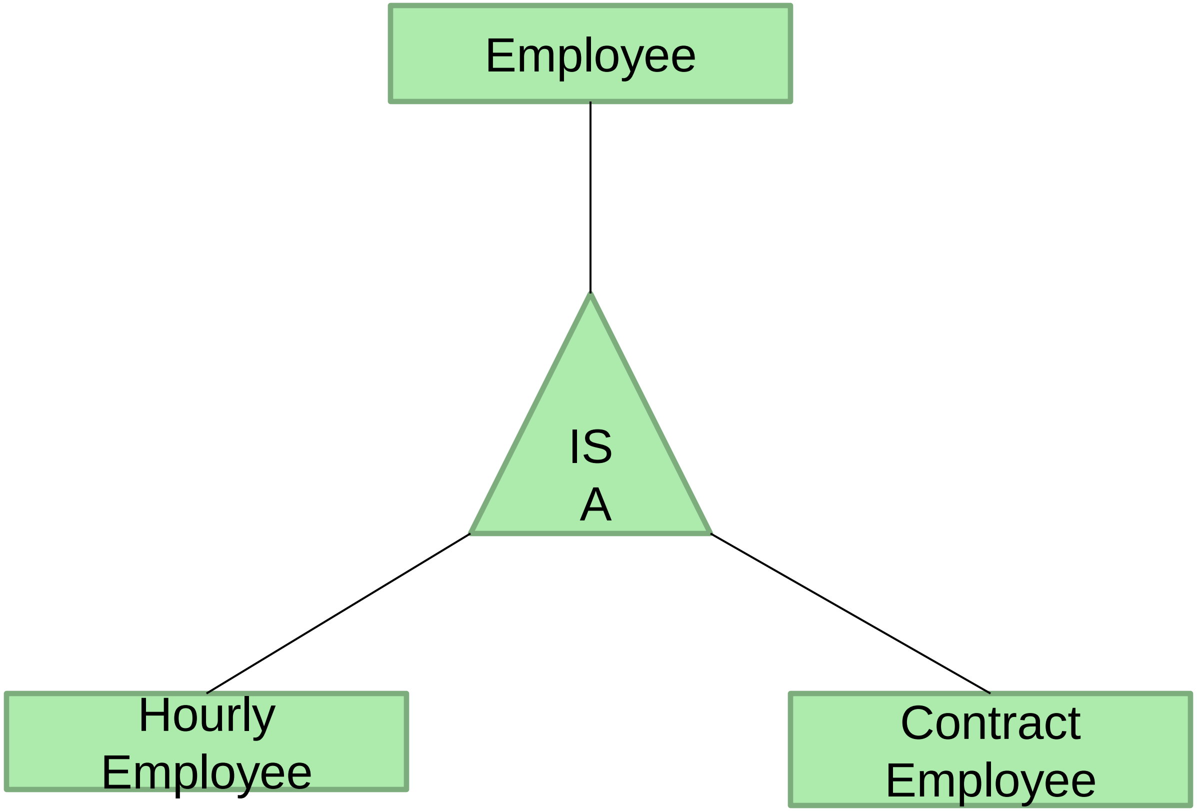
E-R Modeling Participation constraints



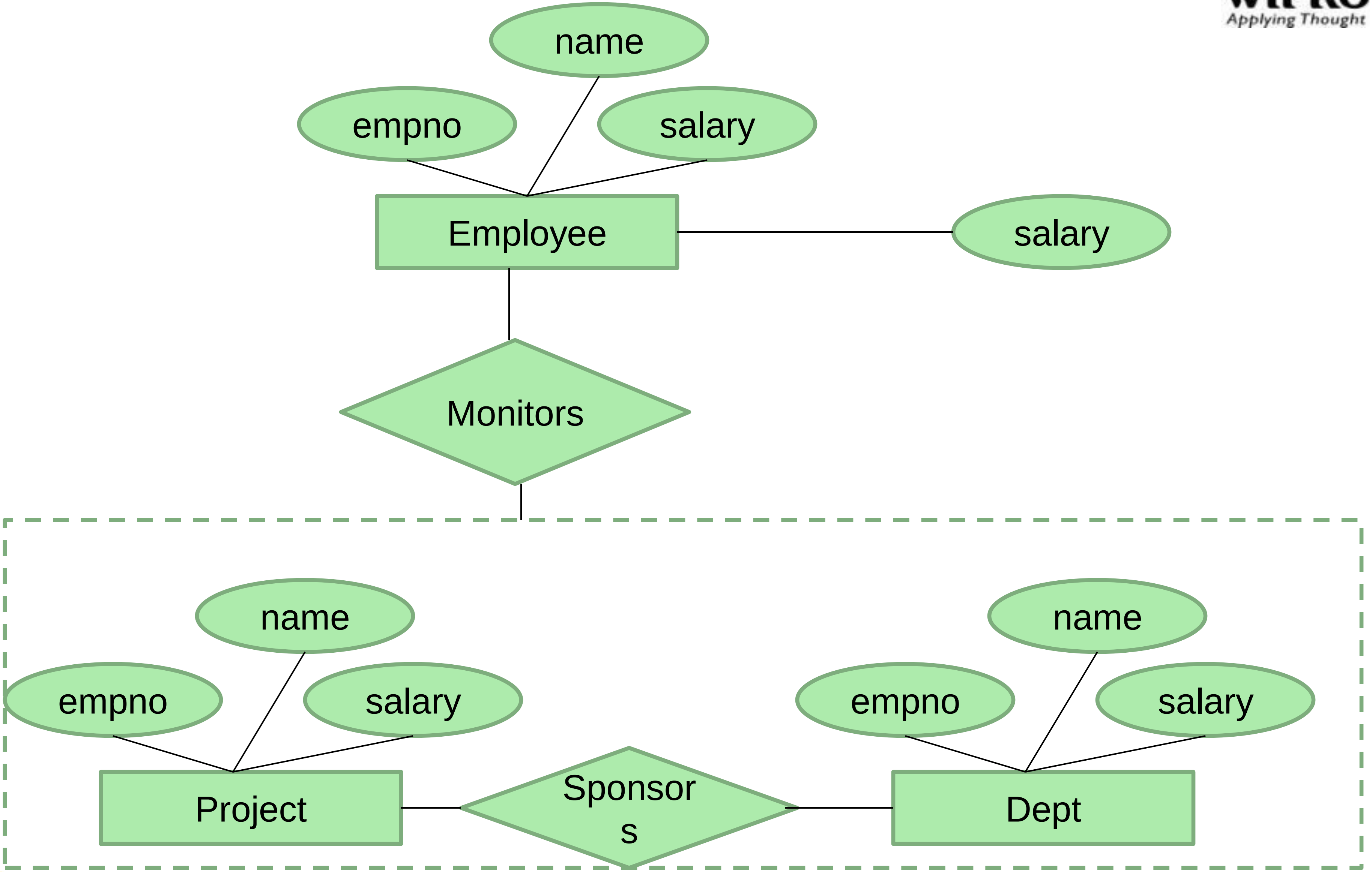
E-R Modeling: Weak entities



E-R Modeling: IS A (‘is a’) hierarchies



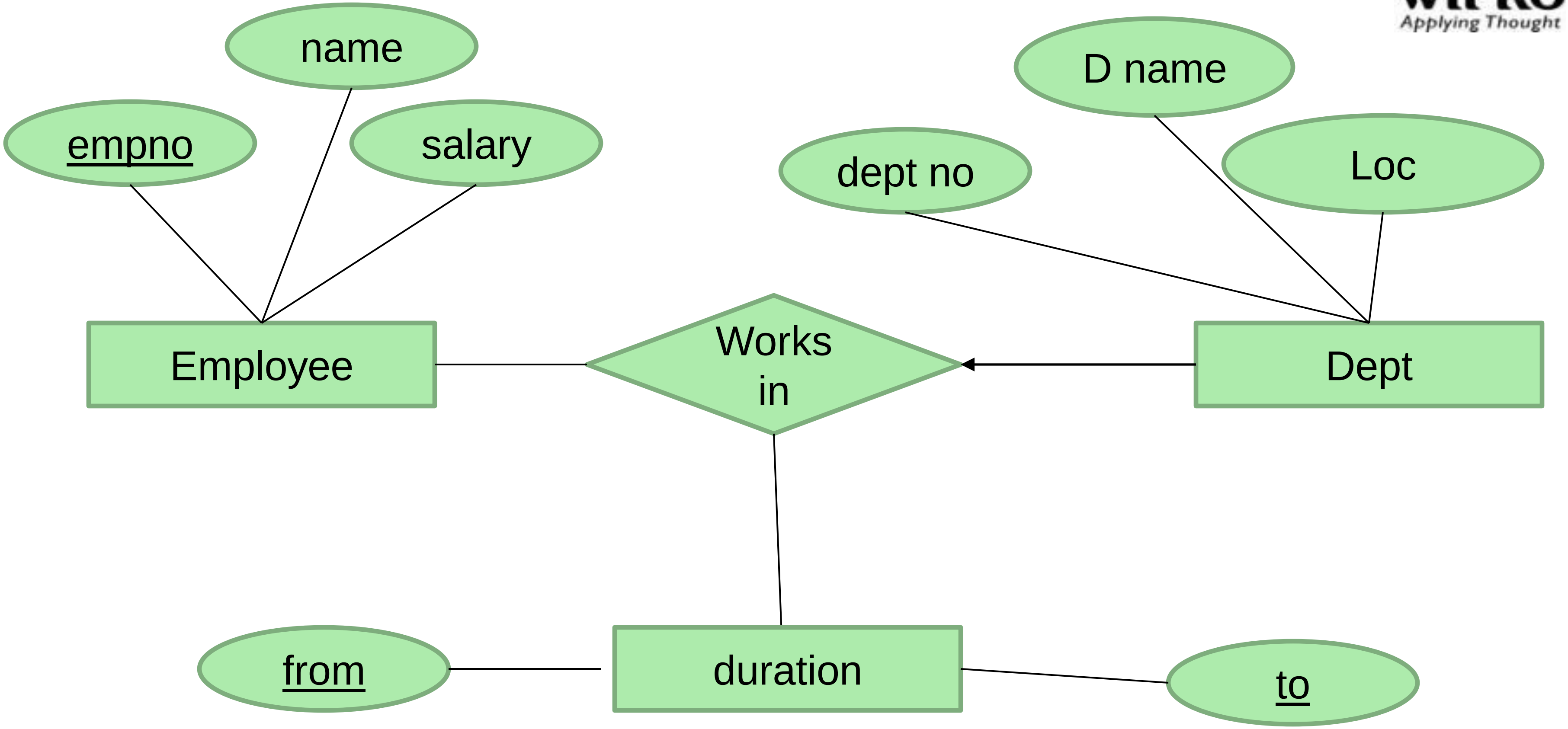
ER-Modeling: Aggregation



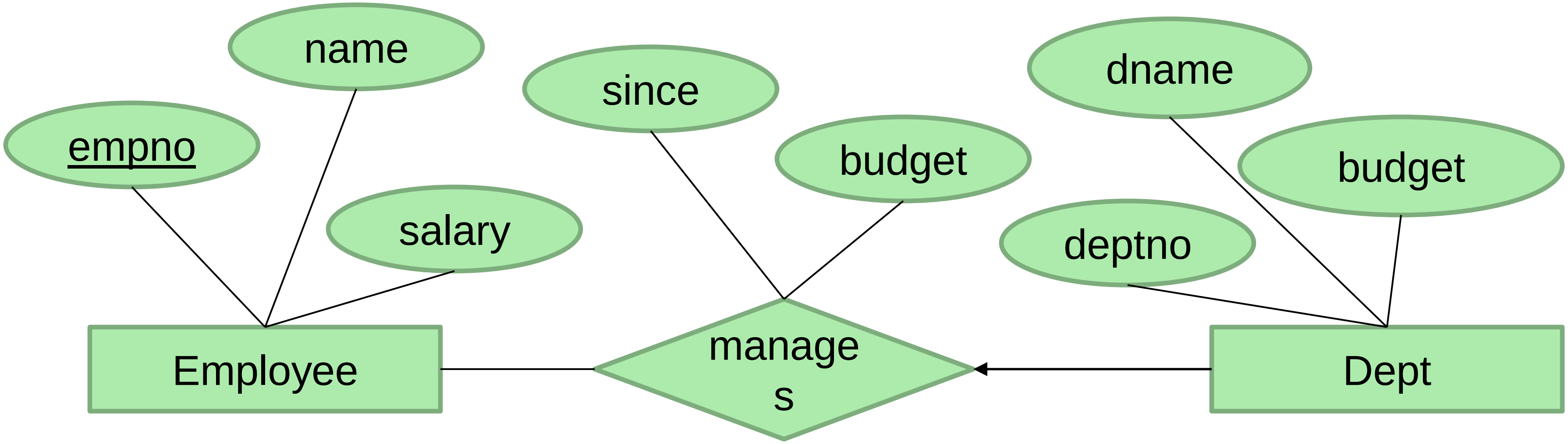
E-R Modeling Entity Vs. Attribute

- It is always an question
- Should address be a attribute of employee entity or a individual entity

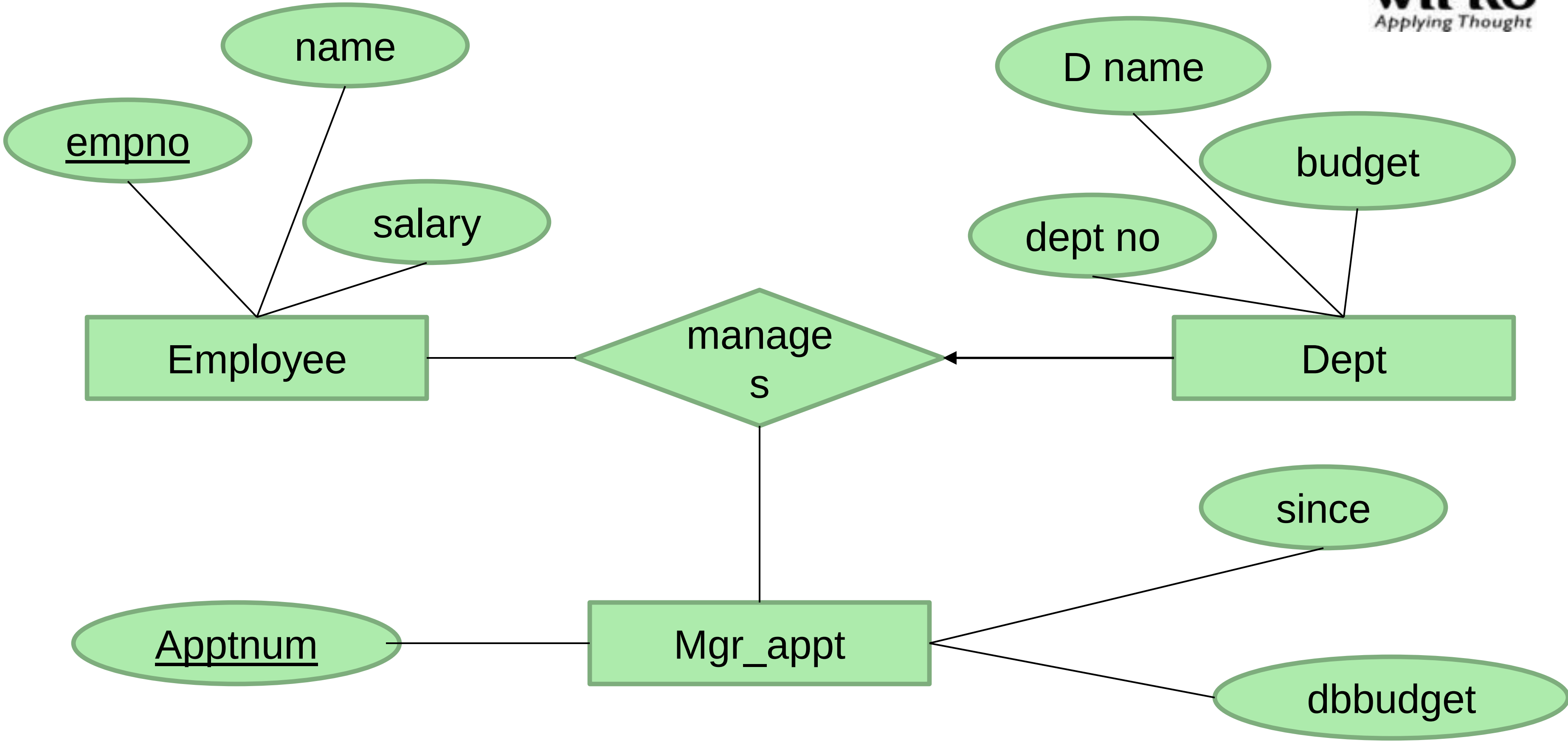
Entity Vs. Attribute (Contd.).



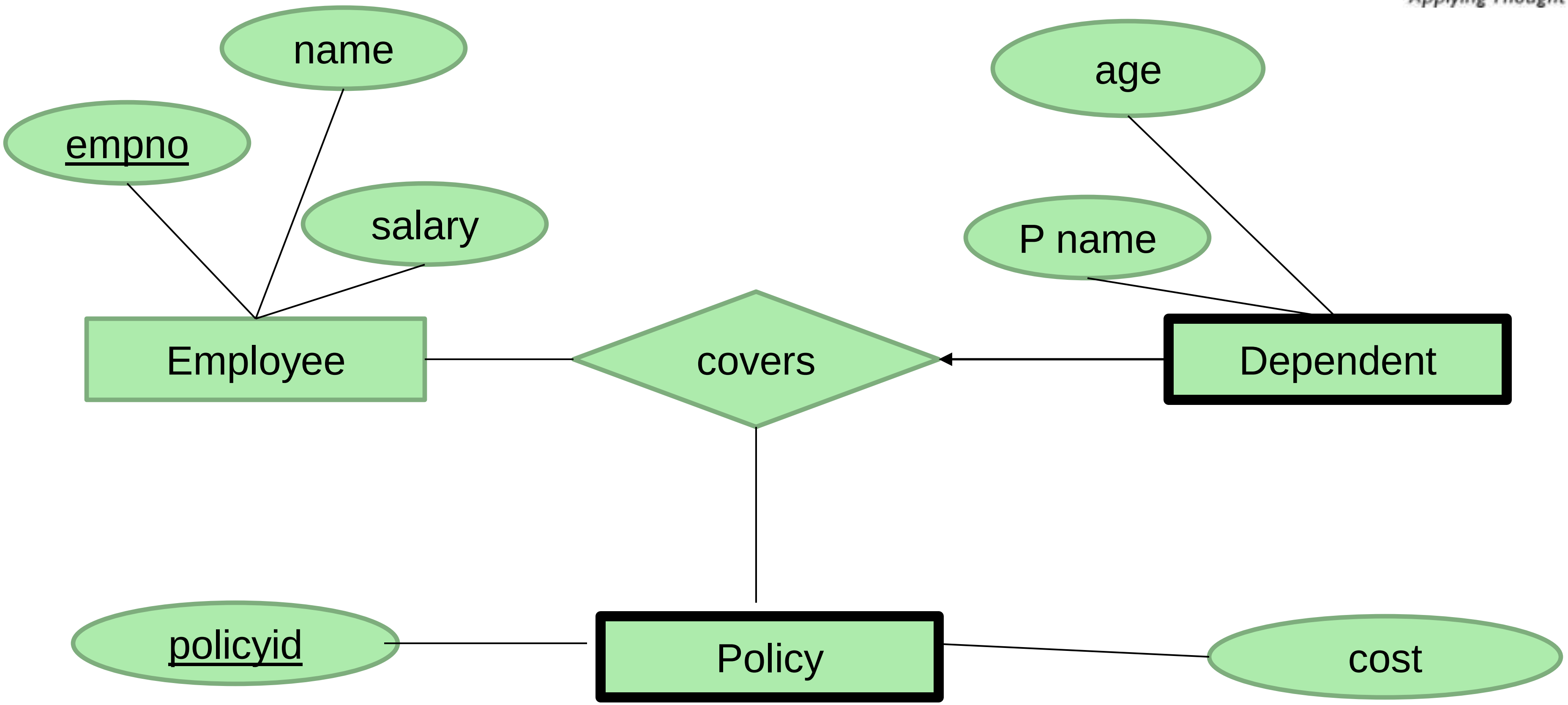
E-R modeling Entity Vs. Relationship



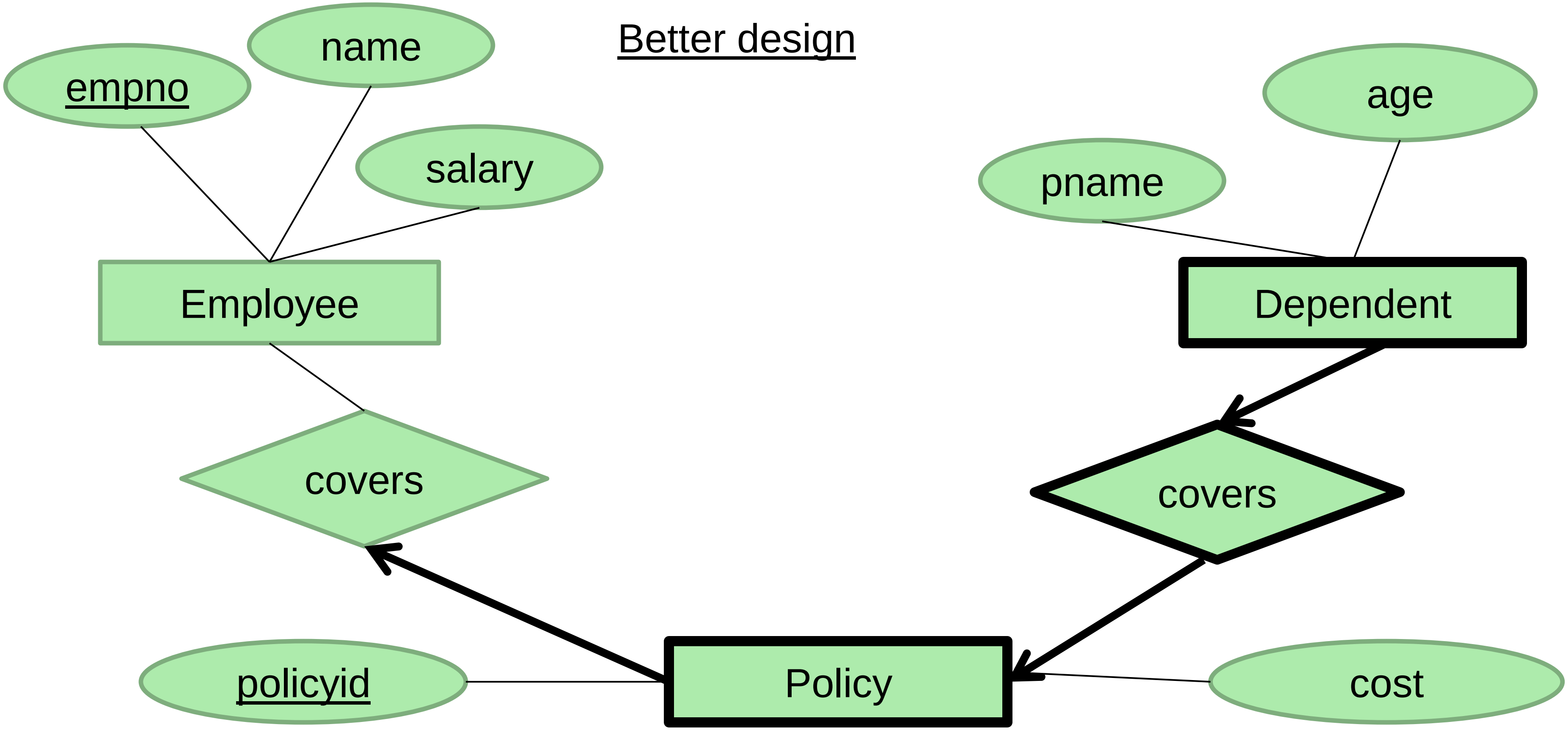
Entity Vs. Relationship (Contd.).



E-R modeling: Binary Vs. ternary Relationships



Binary vs. Ternary relationships (Contd.).



E-R modeling: Constraints beyond the E-R model

- Some constraints cannot be captured in ER Diagrams:
  - Functional dependency
  - Inclusion dependency
  - General constraints

Recap - Conceptual Design

- Conceptual design follows requirements analysis
  - Yields a high – level description of data to be stored
- ER model popular for conceptual design
  - Constructs are expressive, close to the way people think about their applications
- Basic contracts: entities, relationships, and attributes (of entities and relationships)
- Some additional constructs: week entities, IS-A hierarchies, and aggregations
- Note: There are many variations on ER model

Recap - E-R Diagrams

- E-R model and E-R designing phases
- ER modeling notations
- Modeling Entity Relationship diagram
- Even though different kinds of veracity constraints can be shown in the ER model but some cannot be.
- For example check constraint cannot be shown
- We may not be able to show functional dependencies

ER Diagram Hands-on

- Please Refer the Hands-on Reference Document

Schema Refinement & Normalization

Module 3

Objectives

At the end of this module, you will be able to:

- Define Normalization and De-Normalization
- Explain types of normal forms
- Identify problems with un-normalized data
- Perform cleansing an ER diagram

Duration: 3 hrs

Normalizations and normal forms

- What is Normalization:
  - Designing the database without any anomalies
  - Designing the database such that the redundancy is reduced

- Types of Normal forms:
  - First normal form(1NF)
  - 2nd normal form(2NF)
  - 3rd normal form(3NF)
  - Boyce-codd normal form

Why Normal Forms?

- To understand the how complex our data base is
- To know in which state we are in
- This will allow us to understand present criticality in the database
- To ensure that all the operations related to database perform smoothly

Issues with Redundancy

- It is the main root cause for all the anomalies
- Anomalies like INSERT, UPDATE, DELETE
- Huge memory loss

Insert anomalies

Name	Father name	Address	Exam	Subject	marks
Suresh	Abhishek	Bangalore	OCP	SQL	10
Kumar	Rajesh	Mysore	OCA	PLSQL	20

- Consider the above example for the insert anomaly
- If we need insert Suresh record for the next exam then we need to once again enter all the details such as name, father's name and address

Update anomalies

Name	Father name	Address	Exam	Subject	marks
Suresh	Abhishek	Bangalore	OCP	SQL	10
Kumar	Rajesh	Mysore	OCA	PLSQL	20

- Consider the above example for the update anomaly
- if any student requests for the address change we need to change all the records where all the address for the same person

Delete anomalies

Name	Father name	Address	Exam	Subject	marks
Suresh	Abhishek	Bangalore	OCP	SQL	10
Kumar	Rajesh	Mysore	OCA	PLSQL	20

- Consider the above example for the insert anomaly
- If we need to delete any one record then we may also delete the record that may be needed in future
- For example the subjects that are required for OCP will be lost if Suresh record is deleted
- After the record is deleted then we will not be in position what are the subjects that need to be covered in OCP exam

First Normal Form

- If all the values in a relation are of atomic in nature then it can be considered as it is in 1st normal form

Below is an example with shows that it is not in 1st normal form

Name	Father name	Address	Exam	Subject	marks
Suresh	Abhishek	Bangalore	OCP	SQL PLSQL Fundamental 1 Fundamental 2	10
Kumar	Rajesh	Mysore	OCA	PLSQL	20

First Normal Form (Contd.).

- As per the rules of the first normal form below is the example that makes the table in to first normal form

Name	Father name	Address	Exam	Subject	marks
Suresh	Abhishek	Bangalore	OCP	SQL	10
Suresh	Abhishek	Bangalore	OCP	PLSQL	10
Suresh	Abhishek	Bangalore	OCP	Fundamental 1	10
Suresh	Abhishek	Bangalore	OCP	Fundamental 2	10
Kumar	Rajesh	Mysore	OCA	PLSQL	20

Functional dependencies

- Provides a formal mechanism to express constraints between attributes
- Every key in a relation should be functionally dependent on its primary key
- In order to check the functional dependencies
- We need to first make one primary key
- And then check all the other attributes are functionally dependent on it
- If any attribute found which is not satisfying can be removed and associate it with another relations

Full Dependency

- If there is a primary key with combination of more then two keys then full and partial dependency comes into picture

Name	Exam	marks	

- If name and the Exam is a composite primary key then marks is full dependency to the primary key

Partial dependency

- In a relation when more then one attributes are combined and playing a role of primary key some times it may be seen that some attributes may not be completely dependent on the entire primary key. Rather they may fully dependent on one of the attribute

- In such case it is considered as partial dependency

Name	Exam	marks	DOB

- DOB in the above example is completely dependent on name rather then exam
- So it can be considered as partial dependency

Second normal form (2NF)

- It should be in 1st normal form
- All the attributes should have full dependency
- No partial dependency is allowed

Transitive Dependency

- In a relation we need to check, some attributes may transitively dependent on primary key

Empno	Project name	No of hours	Salary

- In this case let us consider both Emp no and project name as composite primary key
- In such case we can see that number of hours an employee worked is completely dependent on primary key. But salary is more dependent on number of hours the employee works
- In such case we can say that salary is depending on number of hours rather then employee number or project name
- So, a transitive dependency exists between salary and number of hours

Third normal form

- Any relation to be in third normal form
- First it should be in 2nd normal form
- And no other attribute should be found as transitive dependent on the primary key
- In this form each attribute in a relation must truly dependent on the primary key not even partially on other attributes

Boyce-codd normal form (BCNF)

- The intention of BCNF is that 3NF does not satisfactorily handle the case of a relation processing two or more composite or overlapping candidate keys.
- A relation is said to be in boyce codd normal form (BCNF) if and only if every determinant is a candidate key.

De-Normalization

- Some times it so happens that we need to compromise with the needs
- As so much of normalizations will lead to multiple joins. As at front end the reports need to be made
- In such case more joins will make the entire system slow
- So in order to satisfy the customer need we need to go for De Normalization
- It is nothing but merging back the table to some extent where we need to get the data quickly
- Yes there may some redundancy which we have to accept

Recap - Schema Refinement

- The database community has developed a series of guidelines for ensuring that databases are normalized. These are referred to as normal forms and are numbered from one (the lowest form of normalization, referred to as first normal form or 1NF) through five (fifth normal form or 5NF) and even higher, as per requirement. In practical applications, you'll often see 1NF, 2NF, and 3NF along with the occasional 4NF
- Decompositions should be carried out and/ or re- examined while keeping performance requirements in mind. De-normalization is needed if:
  - Relations in higher normal form cause the performance problem
  - Majority of queries are data retrieval
  - De-normalization can speed up the data retrieval, and
  - De-normalization does not introduce severe update anomalies

Schema Refinement & Normalization: Hands-on

- Please Refer the Hands-on Reference Document

Summary

In this module, we discussed:

- Normalization and normal forms
- Problems with un-normalized data
- Functional Dependency
- Transitive Dependency
- Partial and Full Dependencies
- De-normalization

Introduction to Oracle SQL

Module 4

Objectives

At the end of this module, you will be able to

- Identify the features of the data dictionary
- Retrieve rows from tables using SELECT Command
- Retrieve rows from tables using alias
- Create a table by using the CREATE TABLE statement
- Identify the rules for naming tables in a database
- Sequence of steps to create a table
- Identify the Oracle data types with their definitions
- Identify the properties & characteristics of constraints
- Alter the table structure & characteristics of an existing column

Duration: 3 hrs

What is there in the Database?

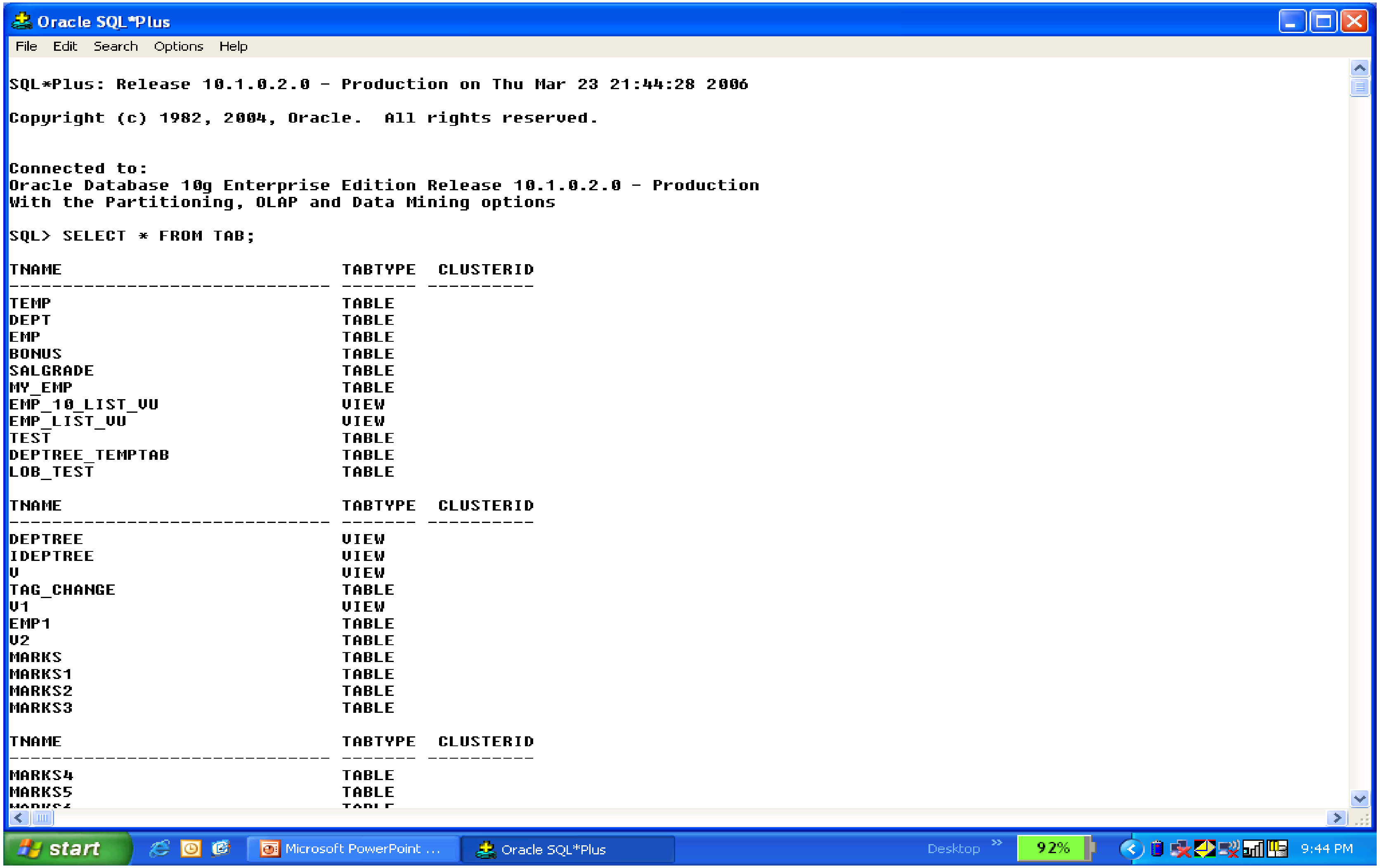
- This will list out all the tables present in scott area (any user who has logged in)

```
SQL>  SELECT * FROM TAB;
```

- What is TAB?
  - A table containing details about all the tables in a user's area

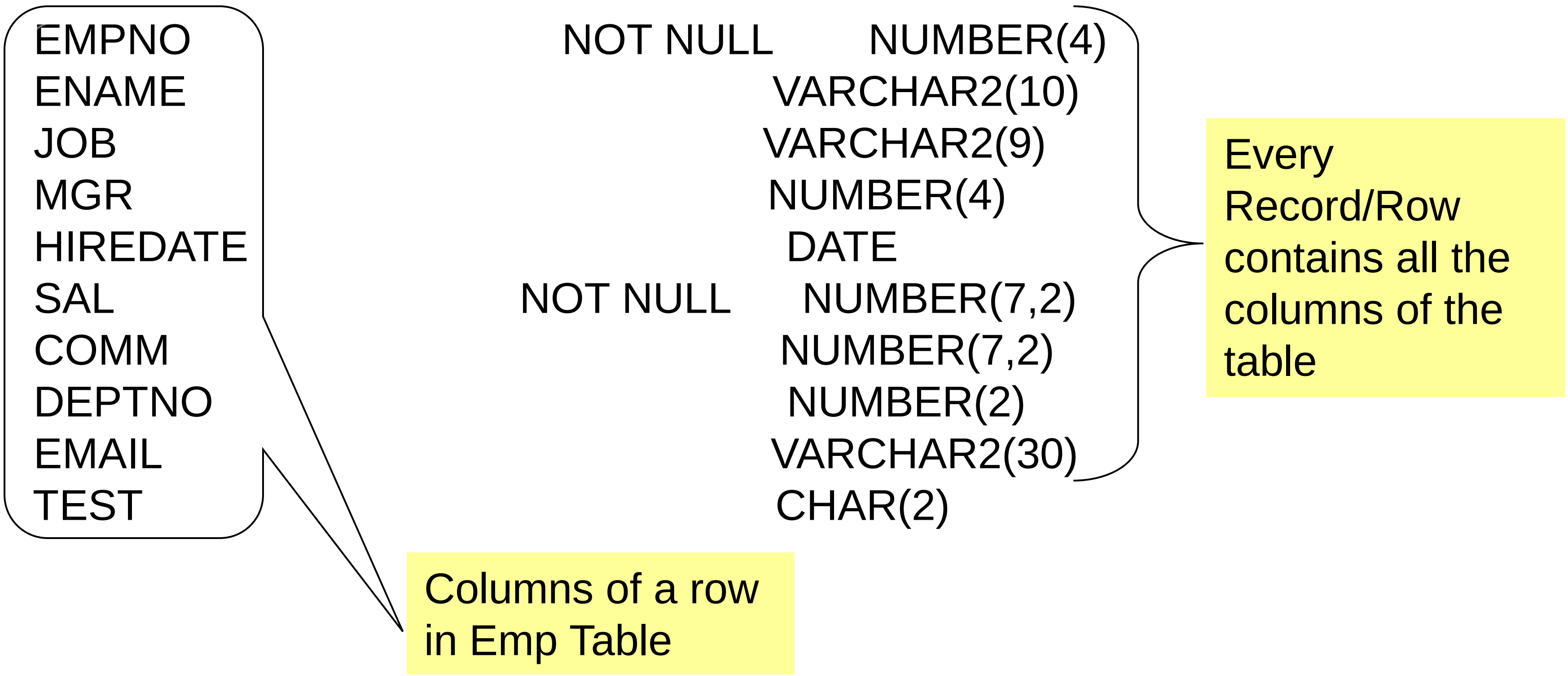
- Anything and everything in the database can be stored only by means of a row in a table

What is there in the database? (Contd.).



```
SQL*Plus: Release 10.1.0.2.0 - Production on Thu Mar 23 21:44:28 2006

Copyright (c) 1982, 2004, Oracle.  All rights reserved.


Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - Production
With the Partitioning, OLAP and Data Mining options

SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
TEMP                           TABLE
DEPT                           TABLE
EMP                            TABLE
BONUS                          TABLE
SALGRADE                       TABLE
MY_EMP                         TABLE
EMP_10_LIST_VU                 VIEW
EMP_LIST_VU                    VIEW
TEST                           TABLE
DEPTREE_TEMPTAB                TABLE
LOB_TEST                       TABLE

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
DEPTREE                        VIEW
IDEPTREE                       VIEW
V                              VIEW
TAG_CHANGE                     TABLE
V1                             VIEW
EMP1                           TABLE
V2                             TABLE
MARKS                          TABLE
MARKS1                         TABLE
MARKS2                         TABLE
MARKS3                         TABLE

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
MARKS4                         TABLE
MARKS5                         TABLE
```

What is a Table?



```
SQL> desc emp
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 EMPNO                                     NOT NULL NUMBER(4)
 ENAME                                              VARCHAR2(10)
 JOB                                                VARCHAR2(9)
 MGR                                                NUMBER(4)
 HIREDATE                                           DATE
 SAL                                       NOT NULL NUMBER(7,2)
 COMM                                               NUMBER(7,2)
 DEPTNO                                             NUMBER(2)
 EMAIL                                              VARCHAR2(30)
 TEST                                               CHAR(2)

SQL>
```

Every Record/Row contains all the columns of the table

Columns of a row in Emp Table

Basic SELECT Statement

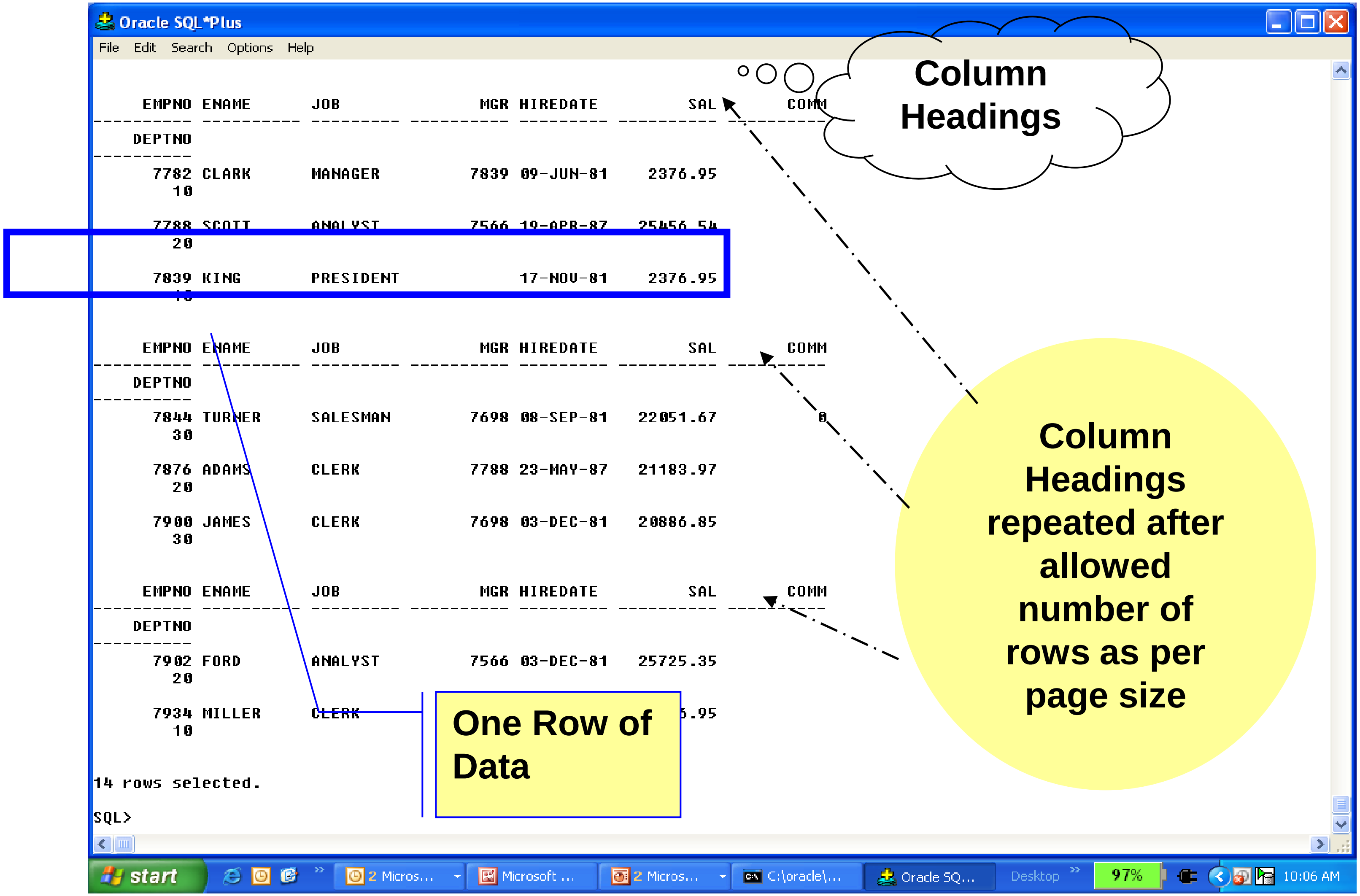
- SELECT statement is the only way of retrieving (querying) stored data from tables

```
SQL>  SELECT * FROM EMP;
```

- * represents all the columns in the mentioned table

- Emp is the table which Oracle has to retrieve the column(s) from

Understanding the Output



```
     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM
---------- ---------- --------- ---------- --------- ---------- ----------
    DEPTNO
----------
      7782 CLARK      MANAGER         7839 09-JUN-81    2376.95
        10

      7788 SCOTT      ANALYST         7566 19-APR-87   25456.54
        20

      7839 KING       PRESIDENT            17-NOV-81    2376.95
        10

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM
---------- ---------- --------- ---------- --------- ---------- ----------
    DEPTNO
----------
      7844 TURNER     SALESMAN        7698 08-SEP-81   22051.67          0
        30

      7876 ADAMS      CLERK           7788 23-MAY-87   21183.97
        20

      7900 JAMES      CLERK           7698 03-DEC-81   20886.85
        30

     EMPNO ENAME      JOB              MGR HIREDATE         SAL       COMM
---------- ---------- --------- ---------- --------- ---------- ----------
    DEPTNO
----------
      7902 FORD       ANALYST         7566 03-DEC-81   25725.35
        20

      7934 MILLER     CLERK                                 5.95
        10

14 rows selected.

SQL>
```

Column Headings

One Row of Data

Column Headings repeated after allowed number of rows as per page size

Selecting only Needed Columns

```
SQL> Select empno,ename,deptno,sal,job
        from Emp;
```

- Selects only the specified columns from the given table
- Columns could be in any order irrespective of the order in which they appear in table definition
- Observe the values appearing in job column

Selecting only Non-Duplicate Rows

- Distinct key word specifies that only unique values from the column(s) selected could be displayed in the output

- Output would be sorted in the ascending order of selected column(s)

```
SQL> Select DISTINCT JOB FROM EMP;
```

Expressions Result in SELECT Statement Output

- Columns to be displayed can also be result of execution of a valid expression

- Arithmetic operators allowed in their order of precedence are * / +-

- Normal precedence can be over-ridden by using parentheses (and)

- Expression in the column list/where clause will get executed for every row

```
SQL> Select empno,sal,sal+100 from emp;
```

Understanding NULL Values

- Wherever comm is NULL, comm*12 is also NULL
- Any expression involving a NULL will result in NULL

```
SQL> Select empno,comm,comm*12
        from emp
```

- NULL is not 0 (zero) and NULL is not blank space, but NULL means undefined quantity
- Codd’s rule emphasize proper treatment of null values in a RDBMS

Column Alias

```
SQL> Select empno,sal,sal*12 as annual_salary,
          comm commission
          from emp;
```

- Annual_salary is the column alias given to the calculated column sal*12 in the output result set
- When writing sub-queries column alias usage becomes mandatory for calculated columns
- ‘as’ key word is optional
- Once aliased, only the aliased name can be used to refer to that column from that result set

Create Table

- Create table command creates a new table (by default) in the present schema

- Table name should not duplicate the existing object name in the same namespace

- Object naming guidelines include:
  - Object name should start with an alphabet
  - Object name can be followed by one or more character/number and special characters in #, _ (underscore) , $
  - Object name should be unique in the namespace allocated for the object
  - Relevantly name the table to reflect its role in the given business context

Data Types

- Column should be of any oracle SQL data type or user defined data type

- Oracle SQL data types:
  - Char
  - Varchar2
  - Date
  - Timestamp
  - Number

Dropping a Table

- Drops the table from the schema and once dropped cannot regain the table again
- All the data stored along with data structure will be lost
- All related constraints defined on the table are dropped
- All the indexes defined on the table will also be dropped

Constraint

- What is a constraint?
  - Restriction to be obeyed by the data when rows are being manipulated in a table

- Why is it needed?
  - To ensure validity and/or meaningfulness of the data

Types of Constraints

- Primary Key constraint
  - NOT Null and Unique values in the column(s) governed
- Unique Key constraint
  - Non duplicate values in the column(s) governed
- Foreign key constraints
  - to protect referential integrity across tables
- Domain Integrity constraints
  - to preserve domain validity of data in a table column
  - To preserve meaningfulness of data in a table column

Altering an Existing Table - 1

Ca_acc#	Ca_cust_id	Ca_acc_type	Ca_open_date	Ca_year_open_bal	Ca_ytd_credit$	Ca_ytd_debit$	Ca_balance$	Ca_status	ca_last_trdate

Purpose: **Add a new column to the cust_accounts table to know the referee's account number and type**

Alter statement

Alter table cust_accounts Add (ca_referee_acc# number(6), ca_referee_acc_type char(2));

Table altered

SQL>

Ca_referee_acc_type	Ca_referee_acc#

Altering an Existing Table - 2

Purpose: **Add a constraint on column ca_status to reflect the legal case bound accounts as ‘LEGAL’ in addition to existing set of statuses which is taken care of by the existing check constraint ck_ca_status.**

Alter statement (to drop existing check constraint)

```
Alter table cust_accounts drop constraint ck_ca_status;

Table altered

SQL>
```

Alter statement (to add new check constraint which includes new conditions and old set as well)

```
Alter table cust_accounts add constraint ck_ca_status check (ca_status IN
(‘ACTIVE’,’BLOCKED’,’LEGAL’) );

Table altered

SQL>
```

Altering an Existing Table - 3

Purpose: **Set today's date as account opening date (ca_open_date) if it is not given by the user/application during row insertion.**

Alter statement (to modify existing column definition)

Alter table cust_accounts modify ca_open_date default sysdate

Table altered

SQL>

Purpose: **Default value of account status should not be 'ACTIVE' and it should be nothing if no value is given during row insertion.**

Alter statement (to remove the default ('ACTIVE') value setting on ca_status column

Alter table cust_accounts modify ca_status default NULL;

Table altered

SQL>

Purpose: **To Rename the column ca_year_open_bal to ca_yo_balance**

Alter statement **Alter table cust_accounts rename column ca_year_open_bal to ca_yo_balance;**

Table altered

SQL>

Restrictions on Altering a Table

- Reducing the column width on a NON empty table is not allowed

- When a new constraint is added to an existing table, constraint will be validated on all existing rows
  - Therefore, adding a column with not null constraint but without default value to it, to a table having some records is not possible

- Dropping of a constraint having dependent constraints on it, is not possible
  - i.e. trying to drop a primary key constraint being referenced by a foreign key constraint

- Newly set default value on a column of a table having records will not be assigned to existing row's respective column

Enabling/ Disabling Constraints

Purpose: **To temporarily disable the foreign key constraint that references primary key of customer table, facilitate fast bulk loading of records onto customer_accounts table (which are having rows with only valid customer_id)**

Alter statement:

```
Alter table cust_accounts disable constraint fk_cust;


Table altered

SQL> -- do all bulk DML operations

SQL> -- Enable the constraint again
SQL> Alter table cust_accounts enable constraint fk_cust;


Table altered

SQL>
```

Enabling/ Disabling Constraints (Contd.).

- Why ?
  - For bulk loading operations in DW kind of databases, to speed up the loading process
  - When constraints are active, will cause a subtle delay in bulk loading operations on the table
  - No need to drop and recreate the needed constraints

Points to Remember

- Data Dictionary
- SELECT Command
- Column Alias
- CREATE TABLE
- Naming conventions of tables in a database
- Oracle data types
- Constraints
- Alter the table structure

DQL & DDL : Hands-on

- Please Refer the Hands-on Reference Document

Scoping and Ordering of Rows

Module 5

Objectives

At the end of this module you will learn be able to:

- Identify Relational operators & expressions
- Identify Logical operators and expressions
- Restrict rows using where clause
- Understand Order by clause and its execution

Duration: 2 hrs

Relational Expression

- Relational expression evaluates to the answer as TRUE or FALSE
  - Marks > 80 true or false
  - Interest < 14 true or false
  - Band = 'A1' true or false

- Relational Operator tests the relation between two scalar quantities

Relational Operators in Oracle

- > < = <> , >= , <=
  - Sal > 1200   deptno <> 10
  - Sal <= 2000   hiredate > ’10-jul-1980’

- IN List operator,

- BETWEEN lower range..AND.. Higher range
  - deptno IN (10,30)
  - Sal between 1000 and 2000

Relational Operators in Oracle (Contd.).

- LIKE - String pattern matching operator
  - % zero or many characters
  - _ (underscore) one character

- IS NULL – Special operator for checking null values
  - COMM IS NULL
  - MGR IS NULL

Logical Operators

- Logical Operators combine logical results (true / false) of two relational expressions (or negates logical result of a relational expression)
  - NOT
  - AND
  - OR

Relational Operators Usage in SQL

EMPNO	ENAME	JOB	DEPT NO	HIREDATE	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	1200	
7788	ARUN	ANALYST	10	19-Apr-87	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81	900	
7844	TURNER	SALESMAN	10	8-Sep-81	2200	0
7876	AKBAR	CLERK	30	23-May-87	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	1800	
7902	CHANDRAN	ANALYST	30	3-Dec-81	1600	
7934	MILLER	CLERK	20	23-Jan-82	800	

```
Select empno,ename,job,deptno,
        hiredate, sal,comm
from emp
```

```
where sal > 2000
```

Relational Operators Usage in SQL (Contd.).

Select empno, ename,job, deptno,

hiredate, sal ,comm

from emp

where sal between 2000 and 2500

EMPNO	ENAME	JOB	DEPTNO	HIREDATE	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	1200	
7788	ARUN	ANALYST	10	19-Apr-87	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81	900	
7844	TURNER	SALESMAN	10	8-Sep-81	2200	0
7876	AKBAR	CLERK	30	23-May-87	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	1800	
7902	CHANDRAN	ANALYST	30	3-Dec-81	1600	
7934	MILLER	CLERK	20	23-Jan-82	800	

Logical Operators Usage in SQL

EMPNO	ENAME	JOB	DEPT NO	HIREDATE	MGR	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	

```
where sal > 1000   AND
hiredate <= ’22-Feb-1981’
```

Logical Operators Usage in SQL (Contd.).

EMPNO	ENAME	JOB	DEPT NO	HIREDATE	MGR	SAL	COM M
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	

where deptno = 20 OR MGR = 7566

Logical Operators Usage in SQL (Contd.).

EMPNO	ENAME	JOB	DEPT NO	HIREDATE	MGR	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	

where NOT SAL between 1000 and 3000

Oracle Specific Operators Usage in SQL

EMPNO	ENAME	JOB	DEPT NO	HIREDATE	MGR	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	

where MGR IN (7566, 7698)

Oracle Specific Operators Usage (Contd.).

EMPNO	ENAME	JOB	DEPTNO	HIREDATE	MGR	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	

where MGR IS NULL

Usage of String Pattern Matching LIKE Operator

EMPNO	ENAME	JOB	DEPTNO	HIREDATE	MGR	SAL	COMM
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	300
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	500
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	1400
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	

WHERE ENAME LIKE ‘A%’

Order of Evaluation of Operators in Expressions

- Arithmetic and string manipulation operators
- All relational operators (including IN, BETWEEN ..AND , LIKE)
- NOT operator will be applied next
- Logical operator AND,
- Logical operator OR

Sorting the Result

- SELECT empno, ename, deptno, hiredate from EMP ORDER BY empno
  - Orders the result records in the ascending order of empno

- SELECT empno, ename, sal, hiredate From EMP ORDER BY hiredate DESC
  - Orders records in the descending order of hiredate (latest date first and earliest date last) prestigious

Sorting Result Based on More Columns

EMPNO	ENAME	JOB	DEPT	HIREDATE	MGR	SAL	COMM
7788	ARUN	ANALYST	10	19-Apr-87	7566	30000	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	0
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2500	1400
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		2100	
7934	MILLER	CLERK	20	23-Jan-82	7782	1200	
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	2500	500
7698	ARJUN	MANAGER	30	1-May-81	7839	2200	
7499	RAM	SALESMAN	30	20-Feb-81	7698	2100	300
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7566	RAGHU	MANAGER	30	2-Apr-81	7839	900	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	800	

Order by deptno, sal desc

Points to Ponder

- Relational operator types
- Order of evaluation of operators in expressions
- Special operator for checking NULL values
- Where clause gets executed for every row of the table and only eligible rows appear in output
- Order by clause is used to sort result of a query

Scope & Ordering Hands On

- Please Refer the Hands On Reference Document

DML Statements

Module 6

Objectives

At the end of this module, you will be able to:

- Identify the DML and transaction control statements
- Insert rows in a table by using the INSERT statement
- Insert into existing table using values of another table with sub query
- Update existing rows in a table by using the UPDATE statement
- Delete rows from a table by using the DELETE statement

Duration: 1hr

INSERT

Syntax

```
INSERT INTO Table_name [ (column1[,column2,…]) ] VALUES (value1[,value2,…]);
```

Example

```
INSERT INTO STOCK ( ITEM_CODE,ITEM_DESC, ITEM_UOM,ITEM_STOCK)
                         VALUES (900267,     ’ALLEN KEY’,  ’NOS’,      12
);
```

Or (if all the column values are provided, no need of mentioning column names)

```
INSERT INTO STOCK
                VALUES (900267,’ALLEN KEY’,  ’NOS’, 12);
```

It is part of ongoing transaction or starts new transaction if given as first command in the session

Using Substitution (&) Variables

- Substitution variable is a SQL*Plus feature, which allows us to enter needed values for any sql statement just before the run time

- Substitution variables are useful for one time data entry into small (reference kind of) tables

- Variable is declared at client side by SQL*Plus

Insert Statement and Sub-query

***Purpose:* To insert output of a query (which may use different tables), a set of records into a target table**

E.g.Wanted to insert records of all employees into Bonus table with following details empno, deptno ,Annual_sal (sal*12), Comm from emp table.

***Select Statement:* SELECT empno,deptno,sal*12 annual_sal, comm from Emp**

***Insert Statement:* INSERT INTO BONUS(empno,deptno,Income,Comm) (SELECT empno,deptno,sal*12 annual_sal, comm from Emp)**

Update Statement

- Update statement updates set of column(s) in (by default) all the rows of the table
- When a ‘where’ clause is specified only rows that satisfy the where clause are updated
- It is part of the ongoing transaction and starts a new transaction if given as first statement

Delete Statement

- Delete statement deletes row(s) in a table
- Without Where clause all rows of the table will be deleted
- Delete statement is part of a transaction

Points to Ponder

- DML statements
- DML operations should obey all the concerned constraints in place.
- Where clause in Update and Delete
- Sub query usage in DMLs

DML : Hands-on

- Please Refer the Hands-on Reference Document

Group Functions, Group by, Having Clause & Joins

Module 7

Objectives

At the end of this module, you will be able to:

- Identify the features of a group function
- Write SQL statements that contain common group functions
- Group rows retrieved by using the GROUP BY clause
- Restrict groups of rows retrieved by using the HAVING clause
- Restrictions Governing Group by clause
- Understand the Execution sequence of Query
- Identify the requirements & characteristics for using sub queries

Duration: 2 hrs

Group Functions & Behavior

- Group functions act only on the selected rows after applying Where clause if present in the select statement
- Grouping column contains a Null value; it will form a separate group and hence there will be one row with Null group column(s) value in the result-set
- Group functions will ignore Null values in the aggregate columns

AVG

COUNT

MAX

MIN

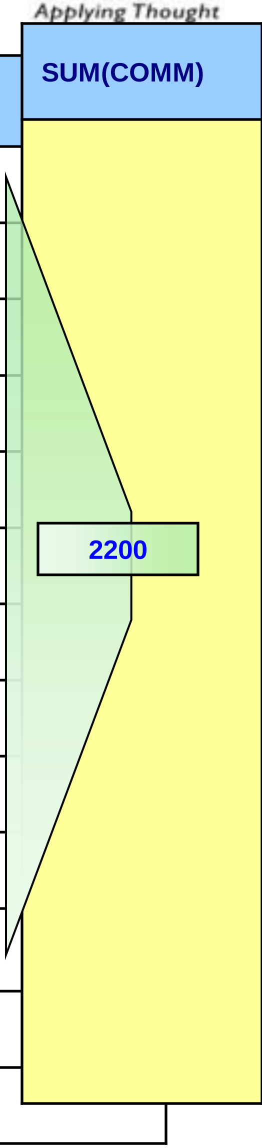
STDDEV

SUM

VARIANCE

Multiple Row SQL Function

EMPNO	ENAME	JOB	DEPTNO	HIREDATE	MGR	SAL	
7499	RAM	SALESMAN	30	20-Feb-81	7698	2200	
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	3000	
7566	RAGHU	MANAGER	30	2-Apr-81	7839	2500	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2100	
7698	ARJUN	MANAGER	30	1-May-81	7839	2500	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	1200	
7788	ARUN	ANALYST	10	19-Apr-87	7566	2500	
7839	RAJ	PRESIDENT	20	17-Nov-81		900	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	2200	
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	1600	
7934	MILLER	CLERK	20	23-Jan-82	7782	800	



When to use Group by Clause?

- In Select Statement by default entire table records together formed as one group by group functions

- When we need aggregations like below, we have to use Group by Clause:
  - Department-wise total salary of employees
  - Month-wise count of employees joined
  - Stock category wise inventory value
  - Account type and branch-wise average balance maintained

Using Group by Clause

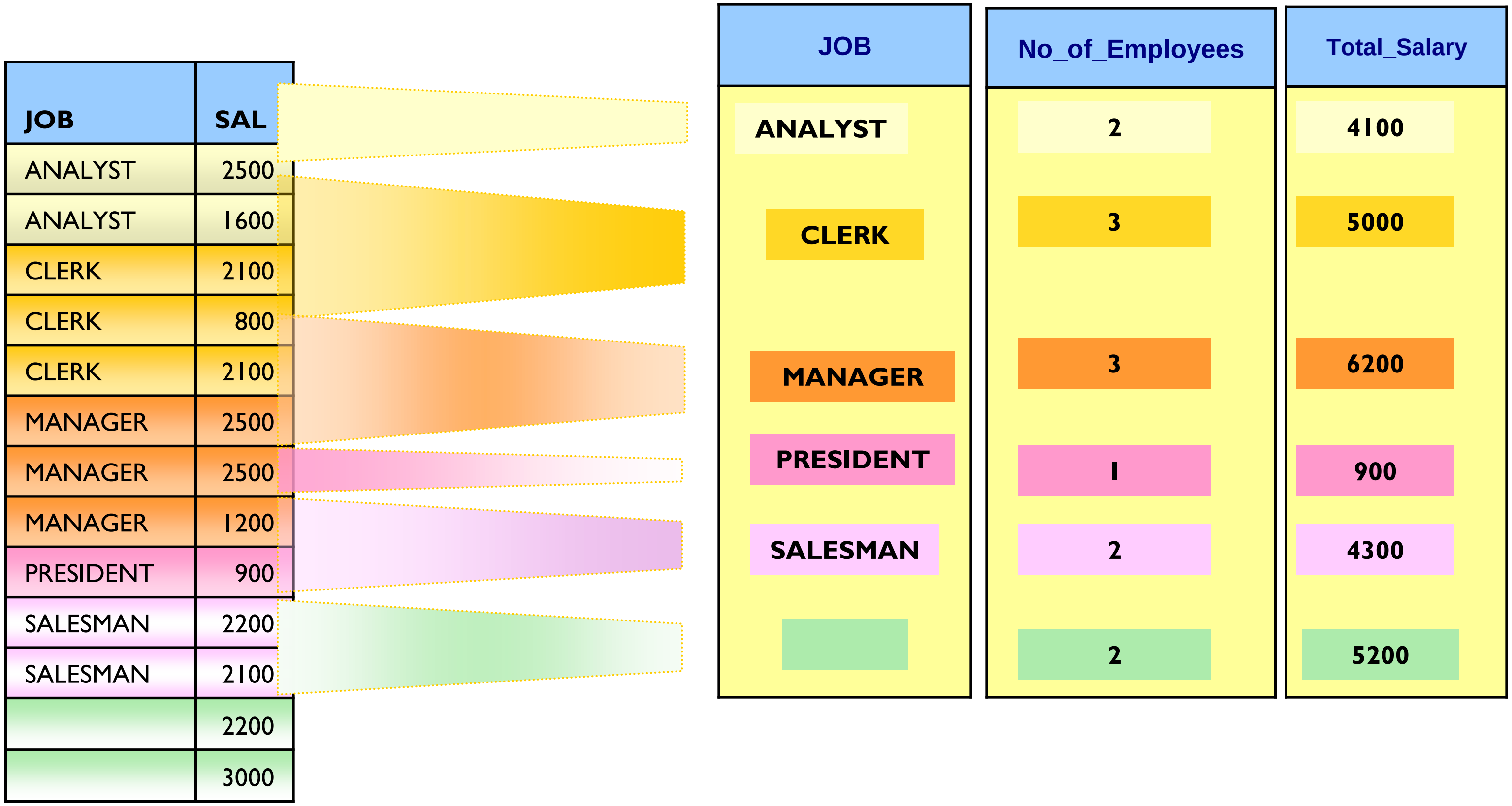
- List out job designation-wise number of employees working, total salary from employees table

Select Statement

```
SELECT JOB,COUNT(*) No_Of_Employees, SUM(SAL) Total_Salary
FROM EMPLOYEES E
GROUP BY JOB;
```

- JOB is the column by which rows of employees table are grouped

Group by Clause Execution



JOB	SAL
ANALYST	2500
ANALYST	1600
CLERK	2100
CLERK	800
CLERK	2100
MANAGER	2500
MANAGER	2500
MANAGER	1200
PRESIDENT	900
SALESMAN	2200
SALESMAN	2100
	2200
	3000

JOB	No_of_Employees	Total_Salary
ANALYST	2	4100
CLERK	3	5000
MANAGER	3	6200
PRESIDENT	1	900
SALESMAN	2	4300
	2	5200

Having Clause

- Where clause is executed on actual rows of the table being queried

- Where clause cannot be used for the following:
  - To filter output records of grouped results satisfying some grouped functions' result(s)
  - To filter out records satisfying some other grouped function (other than group function result sought for) result(s)

Restrictions on Select Statement with Group by Clause

Select Statement Query1

```
SELECT  COUNT(*) NO_OF_EMPS,AVG(SAL) AVG_SAL
FROM     EMPLOYEES
WHERE   SUM(SAL) > 10000
```

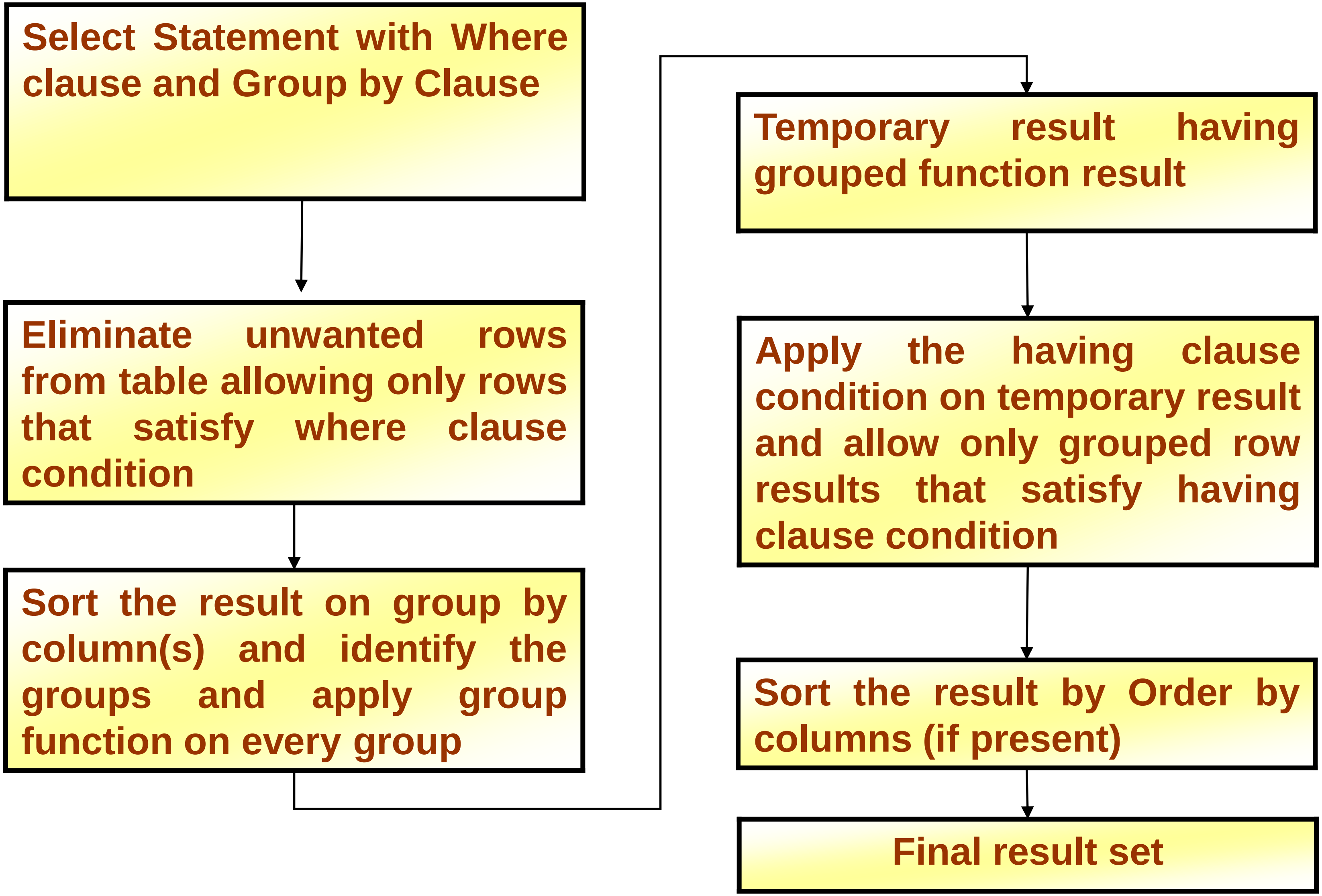
🚫

Select Statement Query2

```
SELECT  DEPT_NO, ENAME,JOB,
             COUNT(*) NO_OF_EMPS,AVG(SAL) AVG_SAL
FROM     EMPLOYEES
WHERE   COMM IS NOT NULL
```

🚫

Execution Sequence of a Complete Query



Select Statement with Where clause and Group by Clause

Eliminate unwanted rows from table allowing only rows that satisfy where clause condition

Sort the result on group by column(s) and identify the groups and apply group function on every group

Temporary result having grouped function result

Apply the having clause condition on temporary result and allow only grouped row results that satisfy having clause condition

Sort the result by Order by columns (if present)

Final result set

Why Join?

- Normalization suggests decomposition of tables to avoid anomalies when manipulating table data

- Reports demand un-normalized data and therefore we go for Joins

What is Join?

- Retrieving data from two tables' related rows between them on the basis of relation between a set of column(s) between them

- Mandatory condition to join two tables is that at least one set of column(s) should be taking values from same domain in each table

Types of Join

- Inner Join (considers only pairs that satisfy the joining condition)
  - Equality Condition

    Table1.column1 = table2.column2
  - Non equality condition

    Table1.column1 > table2.column2
  - Self Join

    Table1.column1 = table1.column2

- Outer Join (Includes also the rows from table(s) even if they do not satisfy the joining condition)
  - Right outer Join
  - Left outer Join
  - Full outer Join

Equi join

Department Table

DEPT_CODE
IMG
BSFI
TMTS
NEW1
NEW2

EmployeesTable

EMPNO	NAME	DEPT_CODE	LOC_CODE
7499	RAM	IMG	BDC
7369	GOPAL	BSFI	BDC
7698	NAREN	TMTS	CDC
6348	VIVEK	BSFI	CDC
7021	JOSEPH	IMG	PDC
7688	RAHEEM	IMG	HDC

Joining Purpose: to List out Department heads and their names

Joining Condition : **Department.dept_Head** **Employees.EMPNO**

Equi join (Contd.).

Cartesian Product

DEPT_HEAD	
7499	
7499	
7499	
7499	
7499	
7499	
6348	
6348	
6348	
6348	
6348	
6348	
7698	7499
7698	7369
7698	7698
7698	6348
7698	7021
7698	7688

Equi join Result

DEPT_CODE	DEPT_HEAD	NAME
IMG	7499	RAM
TMTS	7698	NAREN
BSFI	6348	VIVEK

Non Equi join

Income Tax (Alias T)	High_Ann_Sal	
10000	12000	1
12001		
16001		
22001		

EmployeesTable (Alias E)

EMPNO	NAME	DEPT_CODE	Ann_SAL
7499	RAM	IMG	12000
7369	GOPAL	BSFI	14000
7698	NAREN	TMTS	17000
6348	VIVEK	BSFI	12000
7021	JOSEPH	IMG	15000
7688	RAHEEM	IMG	28000

Joining Purpose: to List out Employees and their respective Income Tax Slab

Joining Condition: **E.Ann_Sal** **T.Low_Ann_Sal** <= **E.Ann_Sal** **T.High_Ann_Sal**

Non Equi join Result

EMPNO	ENAME	DEPT_CODE	Ann_SAL
7499	RAM	IMG	12000
7369	GOPAL	BSFI	14000
7698	NAREN	TMTS	17000
6348	VIVEK	BSFI	12000
7021	JOSEPH	IMG	15000
7688	RAHEEM	IMG	28000

Low_Ann_Sal	High_Ann_Sal	IT_Slab
10000	12000	1
12001	16000	2
16001	22000	3
10000	12000	1
12001	16000	2
22001	99999	4

Select Statement

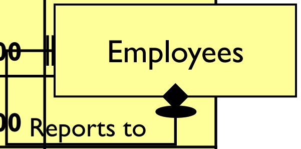
SELECT EMPNO,ENAME,DEPT_CODE,ANN_SAL,Low_Ann_Sal,High_Ann_Sal,IT_SLAB

FROM EMPLOYEES E , ITAX_SLAB T

WHERE E.ANN_SAL BETWEEN T.LOW_ANN_SAL AND T.HIGH_ANN_SAL

Self Join

EMPNO	ENAME	JOB	DEPTNO	HIREDATE	MGR	SAL	COMM
7788	ARUN	ANALYST	10	19-Apr-87	7566	2200	
7844	TURNER	SALESMAN	10	8-Sep-81	7698	3000	0
7934	MILLER	CLERK	20	23-Jan-82	7782	2500	
7900	CAPTAIN	CLERK	20	3-Dec-81	7698	2100	
7654	RAVI	SALESMAN	20	28-Sep-81	7698	2500	1400
7839	RAJ	PRESIDENT	20	17-Nov-81		1200	
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	2500	
7566	RAGHU	MANAGER	30	2-Apr-81	7839	900	
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	2200	
7499	RAM	SALESMAN	30	20-Feb-81	7698	2100	300
7876	AKBAR	CLERK	30	23-May-87	7788	2100	
7698	ARJUN	MANAGER	30	1-May-81	7839	1600	
7521	SHYAM	SALESMAN	30	22-Feb-81	7698	800	500



Joining Purpose: to List out Employee details empno,ename,Job,deptno,hiredate,mgr, Employee's Manager's Name (who is also one among the Employees)

Self Join Result

EMPNO	ENAME	JOB	DEPTNO	HIREDATE	MGR	MANAGER
7788	ARUN	ANALYST	10	19-Apr-87	756	
7844	TURNER	SALESMAN	10	8-Sep-81	769	
7934	MILLER	CLERK	20	23-Jan-82	778	
7900	CAPTAIN	CLERK	20	3-Dec-81	769	ARJUN
7654	RAVI	SALESMAN	20	28-Sep-81	7698	ARJUN
7839	RAJ	PRESIDENT	20	17-Nov-81		
7782	JOSEPH	MANAGER	30	9-Jun-81	7839	RAJ
7566	RAGHU	MANAGER	30	2-Apr-81	7839	RAJ
7902	CHANDRAN	ANALYST	30	3-Dec-81	7566	RAGHU
7499	RAM	SALESMAN	30	20-Feb-81	7698	ARJUN
7876	AKBAR	CLERK	30	23-May-87	7788	ARUN
7698	ARJUN	MANAGER	30	1-May-81	7839	RAJ
					698	ARJUN

Employees table has been imitated as two different tables to be joined.

MGR column assumes the same domain of values as that of EMPNO Column.

Select Statement

SELECT E.EMPNO,E.ENAME,E.JOB, E.DEPTNO,E.HIREDATE,E.MGR,M.ENAME as MANAGER
FROM EMPLOYEES E , EMPLOYEES M
WHERE E.MGR = M.EMPNO

Left Outer Join

Department Table

DEPT_CODE	DEPT_HEAD
IMG	7499
BSFI	6348
TMTS	7698
NEW1	
NEW2	

Employees Table

EMPNO	NAME	DEPT_CODE	LOC_CODE
7499	RAM	IMG	BDC
7369	GOPAL	BSFI	BDC
7698	NAREN	TMTS	CDC
6348	VIVEK	BSFI	CDC
7021	JOSEPH	IMG	PDC
7688	RAHEEM	IMG	HDC

Joining Purpose: List department wise employee details, including the departments without any employees in it too

Left Outer Join Result

Dept Left outer join Employees

DEPT_CODE	EMPNO	NAME	LOC_CODE
BSFI	7369	GOPAL	BDC
BSFI	6348	VIVEK	CDC
IMG	7499	RAM	BDC
IMG	7021	JOSEPH	PDC
IMG	7688	RAHEEM	HDC
NEW1	Null	Null	Null
NEW2	Null	Null	Null
TMTS	7698	NAREN	CDC

Employee table columns are Null as there are no employees

```
Select D.Dept_Code,Empno,Name,Loc_Code
from Dept Left Outer join Employees E
on D.dept_code = E.dept_code
Order by D.Dept_Code
```

What is a Sub-query?

- Sub-query is a query written in some part of another query where it is syntactically accepted

- In simple terms, sub-query is a query within another query

Why Sub-query?

Purpose

To list out empno, name, sal and department of those employees who work for department 10 and whose salary is above the average salary of the department employees

Select Statement

```
SELECT  empno,ename,sal,deptno
FROM     emp
WHERE   deptno = 10 AND
              SAL >   (average_salary_of_department_employees)
```

```
SELECT AVG(SAL) FROM EMP WHERE DEPTNO = 10
```

Points to Ponder

- Group functions behavior
- Group by Clause restrictions
- Having clause usage Vs Where clause
- Order of execution of SELECT statement with all clauses
- Select statement without joining condition will end up giving Cartesian product
- Self join is imitating same table for different roles (use appropriate Table alias)
- Joining conditions

Group Function, Group By & Joins Hands-on

- Please refer the Hands-on Reference Document

References

- Department of Computer Engineering, Middle East Technical University (METU)(1967). *Entity- Relationship Model*. Retrieved on, January 3, 2011, from, www.ceng.metu.edu.tr

- Elmasri and Navathe. Fundamentals of Database Systems, Ed 4. New Delhi: Addison Wesley, 2003.

Thank You