Savitribai Phule Pune University

F. Y. B. C. A. (Science) Semester-II

BCA-128 Lab Course – IV

RDBMS Workbook

Name: ____

College Name:____

Roll No.:________________Division:___________________

Academic Year:

Table of Contents

Assignment Completion Sheet

Lab Course II			
Advanced Relational Database Management System Assignments			
Sr. No.	**Assignment Name**	**Marks (out of 5)**	**Instructor Sign**
1	Case study – ER diagram		
2	Case study – ER diagram (with generalization)		
3	Case study – ER diagram (with aggregation)		
4	Using PostgreSQL (demo of PostgrSQL)		
5	Data Definition queries (Create)		
6	Data Definition queries (Alter)		
7	Data Definition queries (Drop)		
8	Simple queries (Select)		
9	Queries with join		
10	Aggregate queries (Group by and Having)		
11	Nested Queries		
12	Data Manipulation queries (Insert)		
13	Data Manipulation queries (Delete)		
14	Data Manipulation queries (Update)		
Total (Out of 70)			
Total (Out of 15)			

CERTIFICATE

This is to certify that Mr./Ms.____

has successfully completed the RDBMS laboratory course in year

__________ and his/her seat no is ____________. He/She has scored

_____ Marks out of 15.

Instructor H.O.D. / Coordinator

Internal Examiner External Examiner

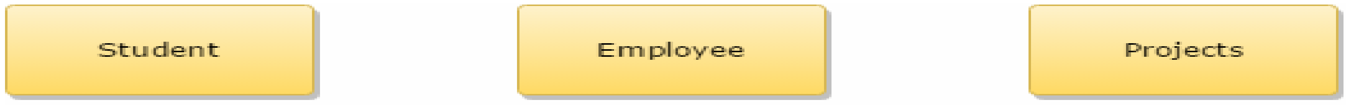
Assignment No. 1

What is Entity RelationshipDiagram (ER-Diagram)?

ER-Diagram is a pictorial representation of data that describes how data is communicated and related to each other. Any object, such as entities, attributes of an entity, sets of relationship and other attributes of relationship can be characterized with the help of the ER diagram.

Entities:

They are represented using the rectangle shape box. These rectangles are named with the entity set they represent.



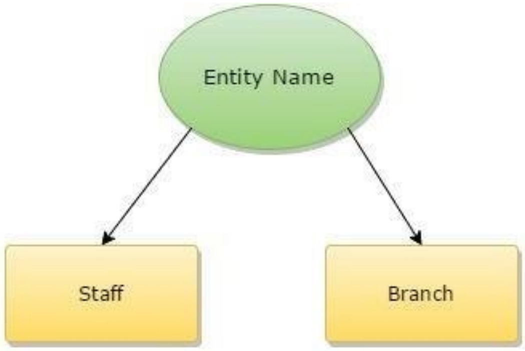
Entities

ER modeling is a top-down structure to database design that begins with identifying the important data called entities and relationships in combination between the data that must be characterized in the model. Then database model designers can add more details such as the information they want to hold about the entities and relationships which are the attributes and any constraints on the entities, relationships, and attributes. ER modeling is an important technique for any database designer to master and forms the basis of the methodology.

- **Entity type:** It is a group of objects with the same properties that are identified by the enterprise as having an independent existence. The basic concept of the ER model is the entity type that is used to represent a group of 'objects' in the 'real world' with the same properties. An entity type has an independent existence within a database.
- **Entity occurrence:** A uniquely identifiable object of an entity type.

Diagrammatic Representation of Entity Types

Each entity type is shown as a rectangle labeled with the name of the entity, which is normally a singular noun.

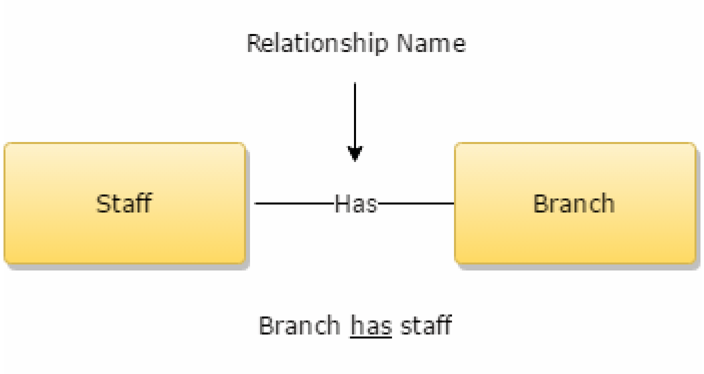


Diagrammatic representation of the Staff and Branch entity types.

What is Relationship Type?

A relationship type is a set of associations between one or more participating entity types. Each relationship type is given a name that describes its function.

Here is a diagram showing how relationships are formed in a database.



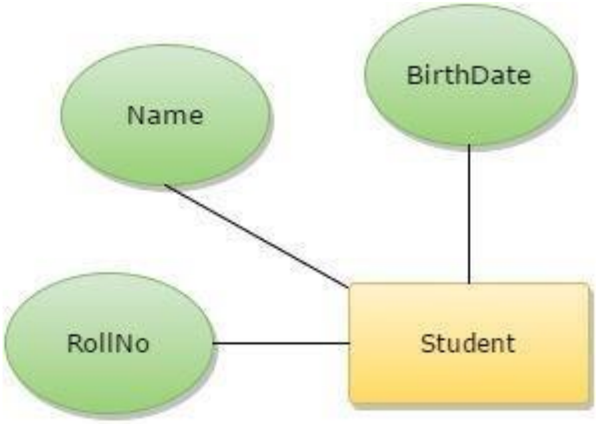
What is degree of Relationship?

The entities occupied in a particular relationship type are referred to as participants in that relationship. The number of participants involved in a relationship type is termed as the degree of that relationship.

In the above figured example "Branch has staff", there is a relationship between two participating entities. A relationship of degree two is called binary degree (relationship).

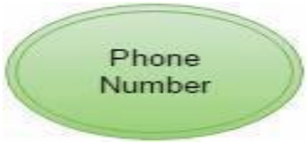
What are Attributes?

Attributes are the properties of entities that are represented by means of ellipse shaped figures. Every elliptical figure represents one attribute and is directly connected to its entity (which is represented as rectangle).



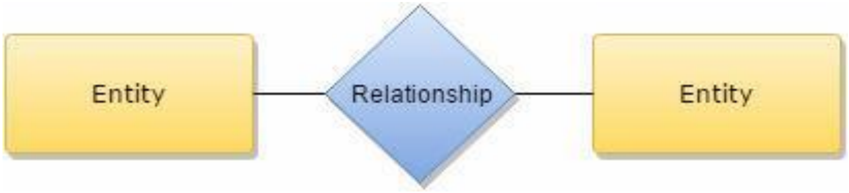
Attributes

It is to be noted that multi-valued attributes are represented using double ellipse like this:



Relationships

Relationships are represented by diamond-shaped box. All the entities (rectangle shaped) participating in a relationship gets connected using a line.



There are four types of relationships. These are:

- **One-to-one:** When only a single instance of an entity is associated with the relationship, it is termed as ‘1:1’.
- **One-to-many:** When more than one instance of an entity isrelated and linked with a relationship, it is termed as ‘1:N’.
- **Many-to-one:** When more than one instance of entity is linked with the relationship, it is termed as ‘N:1’.
- **Many-to-many:** When more than one instance of an entity on the left and more than one instance of an entity on the right can be linked with the relationship, then it is termed as N:N relationship.

Set A

Suppose you are given the following requirements for a simple database for the National Hockey League (NHL):

- the NHL has many teams,
- each team has a name, a city, a coach, a captain, and a set of players,
- each player belongs to only one team,
- each player has a name, a position (such as left wing or goalie), a skill level, and a set of injury records,
- a team captain is also a player,
- a game is played between two teams (referred to as host_team and guest_team) and has a date (such as May 11th, 1999) and a score (such as 4 to 2).

Consider the case study given above and find out entities and their attributes.

Set B

Find different set of entities and their attributes for online bookstore

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2: Late Complete [ ]

3: Needs Improvement [ ] 4: Complete [ ] 5: WellDone [ ]

Signature of Teacher

Assignment no 2

The ER model supported with additional semantic concepts is called the Enhanced Entity-Relationship (EER) model. There are three of the most important and useful added concepts of the EER model, namely specialization/generalization, aggregation, and composition. In this chapter you will learn about the main two important concepts. These are:

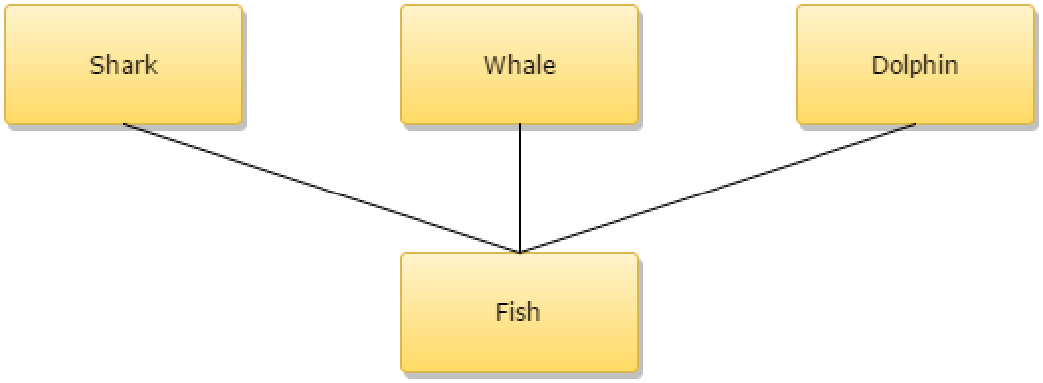
- Generalization Aggregation

What are Generalization / Specialization?

The concept of generalization (specialization) is associated with special types of entities known as super classes and subclasses, and the process of attribute inheritance. Database managers begin this section by defining what super classes and subclasses are and by examining super class/subclass relationships. The ER Model has the capability of articulating database entities in a conceptual hierarchical manner. As the hierarchy goes up, it generalizes the view of entities and as you go deep in the hierarchy, it will provide with the detail of every entity included. Going up in this structure is called generalization, where entities are associated together to represent a more generalized view. Generalization is a bottom-up approach.

In generalization, a number of entities are accommodated together into one generalized entity or category based on their similar characteristics. In the below mentioned figure, whale, shark and dolphin are generalized as fish, i.e. they have been categorized as the fish.

- **Super-class**: An entity type that includes one or more dissimilar sub-groupings of its occurrences that is required to be represented in a data model.
- **Sub-class**: A distinct sub-grouping of occurrences of an entity type that require to be represented in a data model.



Generalization

Super-class/Subclass Relationships

Each member of a sub class is also a member of the super class i.e. the entity in the sub class is the same entity in the super class, but has a different role. The relationship between a super class and a sub class is one-to-one (1:1) and is termed as a super-class/sub-class relationship.

Set A

Refer to the case studies given in assignment no 1

1. Find the relationships among all entities from setA
2. Find the relationships among all entities from setB

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2: Late Complete [ ]

3: Needs Improvement [ ] 4: Complete [ ] 5: WellDone [ ]

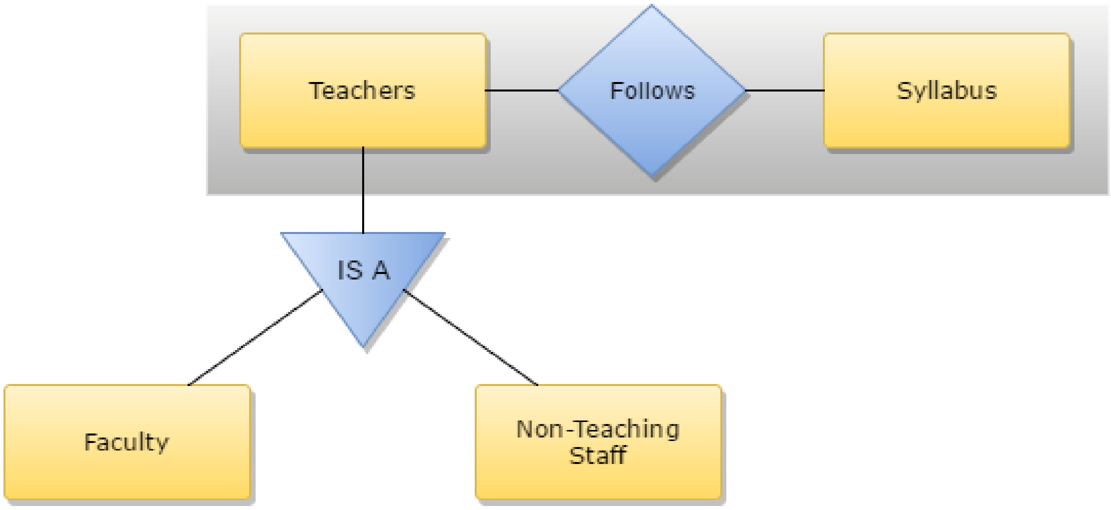
Signature of Teacher

Assignment no.3

What is Aggregation?

A relationship represents a connection between two entity types that are conceptually at the same level. Sometimes you may want to model a ‘has-a’, ‘is-a’ or ‘is-part-of’ relationship, in which one entity represents a larger entity (the ‘whole’) that will consist of smaller entities (the ‘parts’). This special kind of relationship is termed as an aggregation. Aggregation does not change the meaning of navigation and routing across the relationship between the whole and its parts.

An example of an aggregation is the ‘Teacher’ entity following the ‘syllabus’ entity act as a single entity in the relationship. In simple words, aggregation is a process where the relation between two entities is treated as a single entity.



Specialization and Aggregation

Set A

Draw an Entity-Relationship diagram for the National Hockey League (NHL):

- the NHL has many teams,
- each team has a name, a city, a coach, a captain, and a set of players,
- each player belongs to only one team,
- each player has a name, a position (such as left wing or goalie), a skill level, and a set of injury records,
- a team captain is also a player,
- a game is played between two teams (referred to as host_team and guest_team) and has a date (such as May 11th, 1999) and a score (such as 4 to 2).

Set B

Consider a database used to record the marks that students get in different exams of different course offerings.
a) Construct an E-R diagram that models exams as entities, and uses a ternary relationship, for the above database.

b) Construct an alternative E-R diagram that uses only a binary relationship between students and course-offerings. Make sure that only one relationship exists between a particular student and course-offering pair, yet you can represent the marks that a student gets in differentexams of a course offering.

Assignment Evaluation

0: Not Done [ ]   1: Incomplete [ ]   2: Late Complete [ ]

3: Needs Improvement [ ]   4: Complete [ ]   5: WellDone [ ]

Signature of Teacher

Assignment no.4

Using Postgresql (Demo of Postgresql)

Installalation of PostgreSQL on Windows

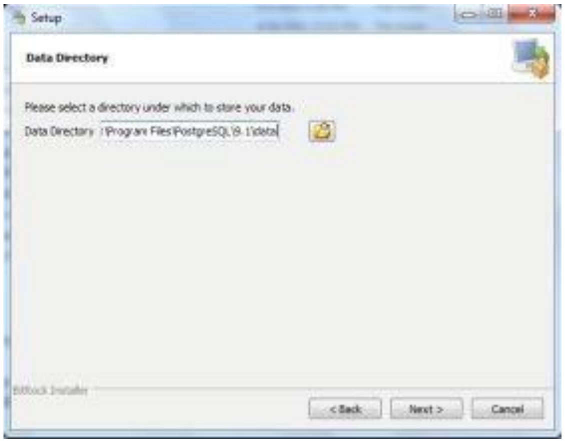
Download PostgreSQL

To download PostgreSQL to install it on Windows 7, please visit the following web page : http://www.postgresql.org/download/windows and click on the "Download" link under "One click installer". The downloaded package will install PostgreSQL Server and pgadmin III GUI to manage PostgreSQL Server and StackBuilder which can be used to download drivers and tools for PostgreSQL Server.

Once you click on the said "Download" link, it will take you to another page from where you need to select the package depending upon your OS platform. So, for installing PostgreSQL on 32 bit Windows 7, select "Win x86-32". If you are using a 64 bit OS, select "Win x86-64". That will start the download process and depending up on your connection speed, take a while to get downloaded.

Make sure you have turned Third Party AntiVirus off while installing.

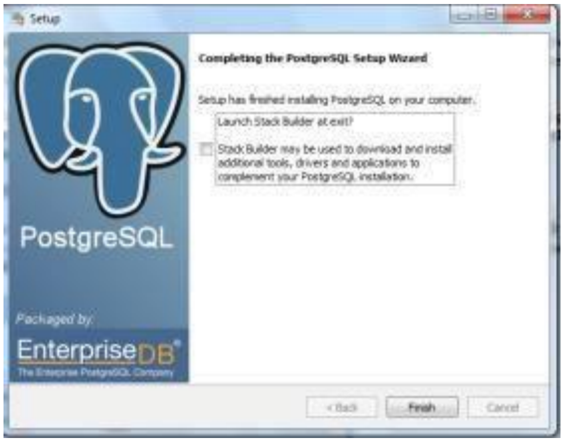
Once the download is finished, run the postgresql-9.1.1-1-windows.exe file and select the location where you want to install it. By default, it is installed within Program Files folder. Then it asks you to enter a password. Keep the port as default. When asked for "Locale", we have selected "English, United States". It will take a while to install PostgreSQL on your system.



Please select a directory under which to store your data.

Data Directory :\Program Files\PostgreSQL\9.1\data

On completion of the installation process, you will get the following screen.



Completing the PostgreSQL Setup Wizard

Setup has finished installing PostgreSQL on your computer.

Launch Stack Builder at exit?

□ Stack Builder may be used to download and install additional tools, drivers and applications to complement your PostgreSQL installation.

After the installation process is completed, you can access pgAdmin III, psql, StackBuilder and PostgreSQL shell from your Program Menu under PostgreSQL 9.1.

Connect to PostgreSQL from the command line

Running the PostgreSQL interactive terminal program, called psql, which allows you to interactively enter, edit, and execute SQL commands. At the time of installing postgres to your operating system, it creates an "initial DB" and starts the postgres server domain running. Typically initdb creates a table named "postgres" owned by user "current logged in user name"

At the command line in your operating system, type the following command.

Debian based systems like Ubuntu :
Connect/login as root -

```
user@user-pc:~$ sudo -i -u postgres
postgres@user-pc:~$ psql
psql (9.3.5, server 9.3.6)
Type "help" for help.
```

Redhat based systems like Centos / Fedora :
Connect/login as root -

```
user@user-pc:~$ su - postgres
user@user-pc:~$ psql
psql (9.3.6)
Type "help" for help.
```

Windows :
In windows, current user doesn't matter

```
C:\Program Files\PostgreSQL\9.4\bin>psql -U postgres
Password for user postgres:
psql (9.4.1)
Type "help" for help.

postgres=#
```

After accessing a PostgreSQL database, you can run SQL queries and more. Here are some common *psql* commands

- To view help for *psql* commands, type \?.
- To view help for SQL commands, type \h.
- To view information about the current database connection, type \conninfo.
- To list the database's tables and their respective owners, type \dt.
- To list all of the tables, views, and sequences in the database, type \z.
- To exit the *psql* program, type \q.

PostgreSQL-Data Types

A datatype specifies, what kind of data you want to store in the table field. While creating table, for each column, you have to use a datatype. There are different categories of data types in PostgreSQL discussed below for your ready reference:

Type	Data Type	Description
Numeric Types	smallint	2-byte small-range integer
	integer, int	A signed, fixed precision 4-byte
	bigint	stores whole numbers, large range 8 byte
	real	4-byte, single precision, floating-point number
	serial	4-byte auto incrementing integer
	double precision	8-byte, double precision, floating-point number
	numeric(m,d)	Where m is the total digits and d is the number of digits after the decimal.
Character Types	character(n), char(n)	Fixed n-length character strings.
	character varying(n), varchar(n)	A variable length character string with limit.
	text	A variable length character string of unlimited length.
Monetary Types	money	currency amount,8 bytes
Boolean type	boolean	It specifies the state of true or false,1 byte.
Date/Time Type	date	date (no time of day),4 byte.
	time	time of day (no date),8 byte
	time with time zone	times of day only, with time zone,12 bytes
	bit(n)	Fixed-length bit string Where n is the length of the bit string.
	varbit(n) bit varying(n)	Variable-length bit string, where n is the length of the bit string.

Assignment no.5

Data Definition Query (Create)

Objective: To create simple tables, with only the primary key constraint (as a table level constraint & as a field level constraint) (include all data types)

A table is a database object that holds data. A table must have unique name, via which it can be referred. A table is made up of columns. Each column in the table must be given a unique name within that table. Each column will also have size a data-type and an optional constraint.

Syntax for **table creation** :

Create tablename(attribute list);
Attribute list : ([attribute name data type optional constraint] , ...)

Create the following tables with primary key constraint

1. Create table emp (eno integer primary key, enamevarchar[50] , salary float);
2. Create table books(id integer UNIQUE, title text NOT NULL, author_idinteger,sub_idinteger,CONSTRAINTbooks_id_pkey PRIMARY KEY(id));
3. Create table sales_order(order_no char[10] PRIMARY KEY, order_date date, salesman_no integer);
4. Create table client_master (client_no integer CONSTRAINT p_client PRIMARY KEY, name varchar[50], addr text, bal_due integer);
5. Create table inventory(inv_no integer PRIMARY KEY, in_stock Boolean);
6. create table sales_order1(order_no char[10], product_no char[10],qty_orderedinteger,product_rate numeric(8,2),PRIMARY KEY(order_no,product_no));

SET A

1. Create a table with following details

Table Name	PLAYER		
Columns	Column Name	Column Data Type	Constraints
1	player_id	Integer	Primary key
2	Name	varchar (50)	
3	Birth_date	date	
4	Birth_place	varchar(100)	
Table level constraint			

2. Create a table with following details

Table Name	Student		
Columns	Column Name	Column Data Type	Constraints
1	Roll_no	integer	
2	Class	varchar (20)	
3	Weight	numeric (6,2)	
4	Height	numeric (6,2)	
Table level constraint	Roll_no and class as primary key		

3. Create a table with details as given below

Table Name	Project		
Columns	Column Name	Column Data Type	Constraints
1	project_id	integer	Primary key
2	Project_name	varchar (20)	
3	Project_ description	text	
4	Status	boolean	
Table level constraint			

4. Create a table with details as given below

Table Name	Donor		
Columns	Column Name	Column Data Type	Constraints
1	Donor_no	integer	Primary key
2	Donor_name	varchar (50)	
3	Blood_group	Char(6)	
4	Last_date	date	
Table level constraint			

Set B

Create table for the information given below by choosing appropriate data types and also specifying proper primary key constraint on fields which are underlined

1. Property (property_id, property_desc , area, rate, agri_status)
2. Actor (actor_id, Actor_name, birth_date)
3. Movie (movie-no, name, release-year)
4. Hospital (hno,hname,hcity)

Set C

Create table for the information given below by choosing appropriate data types and also specifying proper primary key constraint on fields which are underlined

1. Table _______(__________, _____________, ______, _______, Primary key : _________________)

Instructor should fill in the blanks with appropriate values

Assignment Evaluation

0: Not Done [ ]   1: Incomplete [ ]   2: Late Complete [ ]

3: Needs Improvement [ ]   4: Complete [ ]   5: WellDone [ ]

Signature of Teacher

Assignment No.6

Objective: To create one or more tables with Check constraint , Unique constraint, Not null constraint , in addition to the first two constraints (PK & FK)

Constraints can be defined as either of the following:

```
CREATE TABLE table_name
(
column_name1TYPE column_constraint,
column_name2 type column constraint,
table_constrainttable_constraint
);
```

The following are the commonly used column constraints in PostgreSQL:

- NOT NULL – the value of the column cannot be NULL.
- UNIQUE – the value of the column must be unique across the whole table. However, the column can have many NULL values because PostgreSQL treats each NULL value to be unique. Notice that SQL standard only allows one NULL value in the column that has the UNIQUE constraint.
- PRIMARY KEY – this constraint is the combination of NOT NULL and UNIQUE constraints. You can define one column as PRIMARY KEY by using column-level constraint. In case the primary key contains multiple columns, you must use the table-level constraint.
- CHECK – enables to check a condition when you insert or update data. For example, the values in the price column of the product table must be positive values.
- REFERENCES – constrains the value of the column that exists in a column in another table. You use REFERENCES to define the foreign key constraint.

The following are the commonly used table constraints in PostgreSQL:

The table constraints are similar to column constraints except that they are applied to the entire table rather than to an individual column.

The following are the table constraints:

- UNIQUE (column_list)– to force the value stored in the columns listed inside the parentheses to be unique.
- PRIMARY KEY(column_list) – to define the primary key that consists of multiple columns.
- CHECK (condition) – to check a condition when inserting or updating data.
- REFERENCES– to constrain the value stored in the column that must exist in a column in another table.

Syntax for constraints

1. **Null constraint**
   Use of null constraint: Specifies that the column can contain null values
   Use of not null constraint: Specifies that the column can not contain null values
   Ex.: Create table client_master (client_no integer not null, name char(10) not null, addr varchar(30) null, bal_due numeric);

2. **Unique contarint**
Use: forces the column to have unique value.
Ex.: Create table client_master (client_no integer not null, name char(10) not null, addr varchar(30) null, bal_due numeric, ph_no integer unique);

3. **Check constraint**
Use : Specifies a condition that each row in the table must satisfy.Condition is specified as a logical expression that evaluates either true or false.
Ex. Create table client_master (client_no varchar CHECK(client_no like 'C%'), name char(10) check (name=upper(name)), addr varchar(30) null, bal_due numeric, ph_no integer unique);

Set A

1. Create a table with details as given below

Table Name	Machine		
Columns	Column Name	Column Data Type	Constraints
1	Machine_id	integer	Primary key
2	Machine_name	varchar (50)	NOT NULL, uppercase
3	Machine_type	varchar(10)	Type in ('drilling', 'milling', 'lathe', 'turning', 'grinding')
4	Machine_price	float	Greater than zero
5	Machine_cost	float	
Table level constraint	Machine_cost less than Machine_price		

2. Create a table with details as given below

Table Name	Employee		
Columns	Column Name	Column Data Type	Constraints
1	Employee_id	integer	Primary key
2	Employee_name	varchar (50)	NOT NULL, uppercase
3	Employee_desg	varchar(10)	Designation in ('Manager', 'staff', 'worker')
4	Employee_sal	float	Greater than zero
5	Employee_uid	text	Unique
Table level constraint	Employee_uid not equal to Employee_id		

Set B

1. Create a table with details as given below

Table Name	Student		
Columns	Column Name	Column Data Type	Constraints
1	Stud_id	integer	Primary key
2	Stud _name	varchar (50)	NOT NULL, uppercase
3	Stud _Class	varchar(10)	Class in (‘FY’, ‘SY’, ‘TY’)
4	Stud _Marks	float	Greater than zero
5	Stud _uid	text	Unique
Table level constraint	Stud_uid not equal to Stud_id		

Assignment Evaluation

0: Not Done [ ]   1: Incomplete [ ]   2: Late Complete [ ]

3: Needs Improvement [ ]   4: Complete [ ]   5: WellDone [ ]

Signature of Teacher

Assignment No.7

1. Data Definition Queries (Alter) 2. Data Definition Queries (Drop)

Objective: To drop a table from the database, to alter the schema of a table in the Database.

1. **Alter Statement:** Alter table command is use to modify the structure of the table.

Syntax:

ALTER TABLE table_name action;

PostgreSQL provides many actions that allow you to:

- Add a column, drop a column, rename a column, or change a column's data type.
- Set a default value for the column.
- Add a CHECK constraint to a column.
- Rename a table.

The following illustrates the ALTER TABLE statement variants.

1. To **add a new column to a table**, you use **ALTER TABLE ADD COLUMN** statement:

   ALTER TABLE table_name ADD COLUMN new_column_name TYPE;

2. To **remove an existing column**, you use **ALTER TABLE DROP COLUMN** statement:

   ALTER TABLE table_name DROP COLUMN column_name;

3. To **rename an existing column**, you use the ALTER TABLE RENAME COLUMN TO statement:

   ALTER TABLE table_name RENAME COLUMN column_name TO new_column_name;

4. To **change a default value** of the column, you use ALTER TABLE ALTER COLUMN SET DEFAULT or DROP DEFAULT:

   ALTER TABLE table_name ALTER COLUMN [SET DEFAULT value | DROP DEFAULT]

5. To **change the NOT NULL constraint**, you use ALTER TABLE ALTER COLUMN statement:

   ALTER TABLE table_name ALTER COLUMN [SET NOT NULL| DROP NOT NULL]

6. To **add a CHECK constraint**, you use ALTER TABLE ADD CHECK statement:

   ALTER TABLE table_name ADD CHECK expression;

7. To **add a constraint**, you use ALTER TABLE ADD CONSTRAINT statement:

   ALTER TABLE table_name ADD CONSTRAINT constraint_name constraint_definition

8. To **rename a table** you use ALTER TABLE RENAME TO statement:

ALTER TABLE table_name RENAME TO new_table_name;

2. Drop Statement:

Use : Deletes an object (table/view/constraint) schema from the database.
Syntax: drop table table_name;
Example: drop table employee;

Set A

Create the table given below. Assume appropriate data types for attributes. Modify the table, as per the alter statements given below. Type \d <table name> and write the output.
Supplier_master(supplier_no, supplier_name,city,phone-no,amount)

1. Alter table supplier_master add primary key (supplier_no);
2. Alter table supplier_master add constraint city-check check city in('pune', 'mumbai', 'calcutta');
3. alter table supplier_master drop phone-no;
4. alter table supplier_master modify (supplier_namevarchar(50));
5. alter table supplier_master drop constraint city-check;
6. drop table supplier_master;

Set B

1. Remove employee table from your database. Create table employee(eno, ename, sal). Add designation column in the employee table with values restricted to a set of values.
2. Remove student table from your database.
Create table student(student_no, sname,date_of_birth).
Add new column into student relation named address as a text data type with NOT NULL integrity constraint and a column phone of data type integer.
3. Consider the project relation created in the assignment 12. Add a constraint that the project name should always start with the letter 'R'
4. Consider the relation hospital created in assignment 12. Add a column hbudget of type int , with the constraint that budget of any hospital should always > 50000.

Assignment Evaluation

0: Not Done [ ]	1: Incomplete [ ]	2: Late Complete [ ]
3: Needs Improvement [ ]	4: Complete [ ]	5: Well Done [ ]

Signature of Teacher

Assignment No.8

Data Manipulation Queries (Insert, Delete, Update)

Objective: To insert / update / delete records using tables created in previous Assignments. (Use simple forms of insert / update / delete statements)

INSERT syntax

```
INSERT INTO table_name (column1, column2 ...) VALUES (value1, value2 ...);
```

First, you specify the name of the table that you want to insert a new row after the INSERT INTO clause, followed by a comma-separated column list.

Second, you list a comma-separated value list after the VALUES clause. The value list must be in the same order as the columns list specified after the table name.

To add multiple rows into a table at a time, use the following syntax:

INSERT INTO table (column1, column2, ...) VALUES (value1, value2, ...),(value1, value2, ...) ,...;
You just need to add additional comma-separated value lists after the first list, each value in the list is separated by a comma (,).

To insert data that comes from another table, use the INSERT INTO SELECT statement as follows:

INSERT INTO table(value1,value2,...) SELECT column1,column2,... FROM another_table WHERE condition;

The WHERE clause is used to filter rows that allow you to insert partial data from the another_table into the table.

Set A

Consider the tables created in previous assignments .Type and execute the below statements for these tables. Write the output of each statement & justify your answer

1. INSERT INTO sales_order(s_order_no,s_order_date,client_no) VALUES (‘A2100’, now() ,’C40014’);
2. INSERT INTO client_master values(‘A2100’,’NAVEEN’,’Natraj apt’, ’pune_nagar road’,’pune’,40014);
3. Insert into client_master values (‘A2100’,’NAVEEN’,NULL,’pune_nagar road’,’pune’,40014);
4. UPDATE emp SET netsal= net_sal_basic_sal*0.15;
5. UPDATE student
   SET name=’SONALI’,address=’Jayraj apartment’ WHERE stud_id=104 ;
6. DELETE from emp;
7. DELETE from emp WHERE net_sal<1000;

Set B

1. Create the following tables (primary keys are underlined.). Property(pno ,description, area) Owner(oname, address,phone)

An owner can have one or more properties, but a property belongs to exactly one owner . Create the relations accordingly ,so that the relationship is handled properly and the relations are in normalized form (3NF).

a) Insert two records into owner table.
b) insert 2 property records for each owner .
c) Update phone no of “Mr. Nene” to 9890278008
d) Delete all properties from “pune” owned by “ Mr. Joshi”

2. Create the following tables (primary keys are underlined.).
Emp(eno,ename,designation,sal) Dept(dno,dname,dloc)
There exists a one-to-many relationship between emp & dept.Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).
a) Insert 5 records into department table
b) Insert 2 employee records for each department.
c) increase salary of “managers” by 15%;
d) delete all employees of deparment 30;
e) delete all employees who are working as a “clerk”
f) change location of department 20 to ‘KOLKATA’.

3. Create the following tables (primary keys are underlined.)
Sales_order(s_order_no,s_order_date)
Client(client_no, name, address)
A client can give one or more sales_orders , but a sales_order belongs to exactly one client. Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).
a) insert 2 client records into client table
b) insert 3 sales records for each clientchange order date of client_no ’C004’ to 12/4/08
c) delete all sale records having order date before 10th feb. 08
d) delete the record of client “joshi”

Set C
Create the following tables(Primary keys are underlined)
Machine (mno, name, mtype, mcost)
Part (pno, pname, pdesc)
Constraints : Primary Key constraints, machine name not null, foreign key
Machine & Parts are related with one-to-many relationship.
Create the relations accordingly,so that the relationship is handled properly and the relations are in normalized form(3NF) and insert 5 records into each table.

Solve the following queries:
a) Increase the cost of machine by 10%
b) List all parts whose machine cost is greater than 10001.

Assignment Evaluatio

0: Not Done [ ] 1: Incomplete [ ] 2:Late Complete[]

3: Needs Improvement [ ] 4: Complete [ ] 5: Well Done [ ]

Signature of Instructor

Assignment No.9

1.Simple queries **2.Aggregate queries**

Objective: To understand & get a Hands-on practice on Select statement

What is an aggregate function?

An aggregate function produced a single result for an entire group or table.

Aggregate functions are used to produce summarized results. They operate on sets of rows. They return results based on groups of rows. By default, all rows in a table are treated as one group. The GROUP BY clause of the select statement is used to divide rows into smaller groups.

List of aggregate functions

Name	Description
COUNT	This function returns the number or rows or non NULL values for a column
SUM	This function returns the sum of a selected column.
MAX	This function returns the largest value of a specific column.
MIN	This function returns the smallest value of a specific column.
AVG	This function returns the average value for a specific column.

Syntax

aggregate_name (expression [,...] [order_by_clause])

OR

aggregate_name (ALL expression [,...] [order_by_clause])

OR

aggregate_name (DISTINCT expression [,...] [order_by_clause])

OR

aggregate_name (*)

Set A

Create a table employee with attributes empno, name, address, salary and deptno. Insert atleast 10 records into the same. Execute each query

Execute following select queries .

1. Select * from emp;
2. Select empno, name from emp;
3. Select distinct deptno from emp;
4. Select * from emp where deptno =____;
5. Select * from emp where address = ‘pune’ and sal>_______;
6. Select * from emp where address = ‘pune and salary between_______and_____;
7. Select * from emp where name like ‘---%’
8. Select * from emp where name like ‘%and%’

9. Select * from emp where salary is null;
10. Select * from emp order by eno;
11. Select * from emp order by deptno, enodesc;
12. Select deptno as department, sum(salary) as total from emp group by deptno order by deptno;
13. Select deptno as department , count(eno) as total_emp from emp group by deptno having count(eno) >______order by deptno;
14. select avg(salary) from emp;
15. select max(salary),deptno from emp group by deptno having max(sal) >___________;
16. select deptno, min(salary) from emp order by deptno;
17. update emp set salary = salary + 0.5*salary where deptno = (select deptno from department where dname = 'finance');
18. update emp set deptno = (select deptno from department where dname = 'finance')
Where deptno = (select deptno from department where dname = 'inventory');
19. insert into emp_backup(eno,ename) values(select eno,ename from emp);
20. delete from emp where deptno = (select deptno from department where dname='production');

Set B

Prerequisite : Students should know the normalization concept

Consider the relations

Person (pnumber, pname, birthdate, income),

Area (aname,area_type).

An area can have one or more person living in it , but a person belongs to exactly one area.

The attribute 'area_type' can have values as either urban or rural.

Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).

Assume appropriate data types for all the attributes. Add any new attributes as required, depending on the queries. Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries.

Write select queries for following and execute them.

1. List the names of all people living in '_______'area.
2. List details of all people whose names start with the alphabet '_' & contains maximum _ alphabets in it.
3. List the names of all people whose birthday falls in the month of________.
4. Give the count of people who are born on '________'
5. Give the count of people whose income is below________.
6. List names of all people whose income is between_______and______;
7. List the names of people with average income
8. List the sum of incomes of people living in '_____'
9. List the names of the areas having people with maximum income (duplicate areas must be omitted in the result)
10. Give the count of people in each area
11. List the details of people living in '_____' and having income greater than______;
12. List the details Of people, sorted by person number
13. List the details of people, sorted by area, person name
14. List the minimum income of people.
15. Transfer all people living in 'pune' to 'mumbai'.
16. delete information of all people staying in 'urban' area

Set C

Create the following tables(Primary keys are underlined):

Sailors(sid,sname,rate,age)
Boats(bid,bname,colour)
Reserves(sid,bid,date)

Sailors and boats are related many to many.

Create the relations accordingly,so that the relationship is handled properly and the relations are in normalized form(3NF) and insert 5 records into each table.

Draw ER diagram for given relational schema and show normalization.

Solve the following quesries:

a) Find all the sailors with a rating above 8.
b) Find the ages of sailors whose name begins and ends with 'P'.
c) Find name of sailors who have reserved red and green boats.

Assignment Evaluation

0: Not Done [ ] 1: Incomplete [ ] 2:Late Complete[]

3: Needs Improvement [4: Complete [ ] 5: Well Done [ ]

Signature of Instructor

Assignment No.10
Queries with set operations

Objective: To understand & get a Hands-on practice using set operations (union ,intersect and except) with select statement.

1. **Union**
   Use: Returns the union of two sets of values, eliminating duplicates.
   Syntax: <select query> Union<select query>
   Ex.: Select cname from depositor Union Select cname from borrower;
2. **Union all**
   Use: returns union of two sets of values ,retaining all duplicates
   Syntax: <select query> Union all<select query>
   Ex.: Select cname from depositor Union allSelect cname from borrower;

3. **Intersect**
   Use:returns the intersection of two sets of values ,eliminating duplicates
   Syntax: <select query> intersect<select query>
   Ex.: Select cname from depositor intersect Select cname from borrower;

4. **Intersect all**
   Use: returns intersection of two sets of values ,retaining all duplicates
   Syntax: <select query> intersect all<select query>
   Ex.: Select cname from depositor intersect all Select cname from borrower;

5. **Except**
   Use: returns the difference between two sets of values.i.e returns all values of set1 not contained in set2,eliminates duplicates
   Syntax: <select query> except<select query>
   Ex.: Select cname from depositor **Except** Select cname from borrower;

6. **Except all**
   Use: returns the difference between two sets of values.i.e returns all values of set1 not contained in set2, retains all duplicates
   Syntax: <select query> except all<select query>
   Ex.: Select cname from depositor **Except** Select cname from borrower;

Note: To use the INTERSECT operator, the columns that appear in the SELECT statements must follow the rules below:

1. The number of columns and their order in the SELECT clauses must the be the same.
2. The data types of the columns must be compatible.

Set A

Consider the following relations, non-teaching, teaching, department.

One deparment can have one or more teaching & non-teaching staff, but a teaching or non-teaching staff belongs to exactly one department. Hence dno is a foreign key in the both the relations. Create these relations in your database .

Non-teaching (empnoint primary key, name varchar(20), address varchar(20), salary int,dno references department)
Teaching(empnoint primary key, name varchar(20), address varchar(20), salary int,dno references department)

Department (dnoint primary key,dname)

- insert at least 10 records into both the relations.
- type the following select queries & write the output and the business task performed by each query

1. Select empno from non-teaching union select empno from teaching;
2. Select empno from non-teaching union all select empno from teaching;
3. Select name from non-teaching intersect select name from teaching;
4. Select name from non-teaching intersect all select name from teaching;
5. Select name from non-teaching except select name from teaching;

6. Select name from non-teaching except all select name from teaching

Set B

Create the following relations, for an investment firm

emp(emp-id ,emp-name, address, bdate)
Investor(inv-name , inv-no, inv-date, inv-amt)

An employee may invest in one or more investments, hence he can be an investor.But an investor need not be an employee of the firm.

Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).

Assume appropriate data types for the attributes. Add any new attributes , as required by the queries. Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries**.**

Write the following queries & execute them.

1. List the distinct names of customers who are either employees, or investors or both.
2. List the names of customers who are either employees , or investors or both.
3. List the names of employees who are also investors.
4. List the names of employees who are not investors

Set C

Employee (emp_no, emp_name, address, city, birth_date, designation,salary)
Project (project_no, project_name, status)
Department (Dept_no, dept_name, location)
Constraints: Employee designation is either ‘manager’, ‘staff’ , ‘worker’.

There exists a one-to-many relationship between Department and Employee. Many employees can work on many projects controlled by a department. Create the relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF) and insert 5 records into each table.

Solve the following queries:

a) Find the details of employee who is having highest salary.
b) Delete all employees of department 20.
c) List the names and salary of employees sorted by their salary.

Assignment Evaluation

0: Not Done [ ]   1: Incomplete [ ]   2:Late complete[]

3: Needs Improvement [4: Complete [ ]   5: Well Done [ ]

Signature of Instructor

Assignment No.11

Queries &sub-queries, with joining of tables

To understand & practice session on nested queries & sub-queries using join operations.

Sub query:

A sub-query is a select-from-where expression that is nested within another query.

Set membership	the 'in' & 'not in' connectivity tests for set membership & absence of set membership respectively.
Set comparison	the < some, > some, <= some, >= some, = some, <> some are the constructs allowed for comparison. = some is same as the 'in' connectivity. <> some is not the same as the 'not n'i connectivity. Similarly sql also provides < all, >all, <=all, >= all, <> all comparisons. <>all is same as the 'not in' construct.
Set cardinality	The 'exists' construct returns the value true if the argument subquery is nonempty. We can test for the non-existence of tuples in a subquery by using the 'not exists' construct. The 'not exists ' construct can also be used to simulate the set containment operation (the super set). We can write "relation A contains relation B" as "not exists (B except A)".

Set A

Create the following relation in your database (primary keys underlined)

Employee(ename, street, city)

Works(ename, company-name, salary)

Company(company-name, city)

Manages(ename, manager-name)

An employee can work in one or more companies, a company can have one or more employees working in it. Hence the relation 'works' with keyattributes as ename, company-name.

An employee manages one or more employees, but an employee is managed by exactly one employee (a recursive relationship), hence the relation 'manages' with key ename.

Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries**.**

Type the following queries , execute them and give the business task performed by each query

1. select ename from works w where salary >= all (select max(salary) from works));
2. select ename form works w where salary = (select max(salary) from works w1 where w1.company-name = w.company-name));
3. select manager-name from manages where manager-name in(select ename from works where company-name = "_______");
4. select manager-name from manages where manager-name not in(select ename from works where company-name = "_________");
5. select ename from works w where salary > some (select salary from works where company-name not in (select company-name from company where city = "_____"));
6. select ename from employee e where city = (select city from employee e1 , manages m where m.ename = e.ename and m.manager-name = e1.ename);
7. select * from employee where ename in (select manager-name from manages)

8 select city count(*) from employee group by city having count(*) >= all (select count(*) from

employee group by city)
9. select ename from works w where salary <> all (select salary from works where ename<>w.ename);
10. select company-name, sum(salary) from works w groupby company-name having sum(sal) >= all (select sum(sal) from works group by company-name)
11. select ename from employeee where city in('_____','_____');
12. select ename from employee e where city = (select city from company c, works w where w.ename = e.name and c.company-name = w.company-name);

Set B

Create the following relations :

Emp(eno,name,dno,salary)
Project(pno,pname,control-dno,budget)

Each employee can work on one or more projects, and a project can have many employees working in it. The number of hours worked on each project , by an employee also needs to be stored.
Create the Relations accordingly, so that the relationship is handled properly and the relations are in normalized form (3NF).
Assume appropriate data types for the attributes. Add any new attributes , new relations as required by the queries.
Insert sufficient number of records in the relations / tables with appropriate values as suggested by some of the queries**.**
Write the queries for following business tasks & execute them.

1. list the names of departments that controls projects whose budget is greater than .
2.list the names of projects, controlled by department No , whose budget is greater than atleast one project controlled by department No .
3. list the details of the projects with second maximum budget

4. list the details of the projects with third maximum budget.
5. list the names of employees, working on some projects that employee number is working.
6. list the names of employees who do not work on any project that employee number works on
7. list the names of employees who do not work on any project controlled by '______' department
8. list the names of projects along with the controlling department name, for those projects which has atleast employees working on it.
9. list the names of employees who is worked for more than 10 hrs on atleast one project controlled by '_____' dept.
10. list the names of employees , who are males , and earning the maximum salary in their department.
11. list the names of employees who work in the same department as '_____'.
12. list the names of employees who do not live in______or_______**.**

Set C

Execute following queries on the relations mentioned in above case study

1. list the names of projects along with the controlling department name, for those projects which has atleast employees working on it.

2. list the names of employees who is worked for more than 10 hrs on atleast one project controlled by ‘______’ dept.

3. list the names of employees , who are males , and earning the maximum salary in their department.

4. list the names of employees who work in the same department as ‘______’.

5. list the names of employees who do not live in_______or________**.**

Assignment Evaluation

0: Not Done [ ]   1: Incomplete [ ]   2:Late Complete[]

3: Needs Improvement [ ]   4: Complete [ ]   5: Well Done [ ]

Signature of Instructor

Assignment No.12

Queries &sub queries, with joining of table

Execute the following queries on the table created in previous assignments.

Set A

Project-Employee database

Consider the database maintained by a company which stores the details of the projects assigned to the employees.

Following are the tables:

PROJECT (PNO INTEGER, P_NAME CHAR(30), PTYPE CHAR(20),DURATION INTEGER)
EMPLOYEE (ENO INTEGER, E_NAME CHAR (20), QUALIFICATION CHAR (15), JOINDATE DATE)

The relationship is as follows:

PROJECT - EMPLOYEE: M-M Relationship , with descriptive attributes as start_date (date), no_of_hours_worked (integer).

Solve the Queries

1. Find the names of the employees starting with 'A'.
2. Find the details of employees working on the project "System".
3. Find the employee numbers of the employees, who do not work on project "Robotics".
4. Get employee number of the employee, who works on at least one project that employee number '2000' works on.
5. List the names of the first three employees in alphabetical order.
6. Find the names of the employees whose duration is more than three years.

Set B

Bank database

Consider the following database maintained by a Bank. The Bank maintains information about its branches, customers and their loan applications.

Following are the tables:

BRANCH (bid integer, brname char (30), brcity char (10))
CUSTOMER (cno integer, cname char (20), caddr char (35), city (20))
LOAN_APPLICATION (lno integer, lamtrequired money, lamtapproved money, l_date date)

The relationship is as follows:

BRANCH, CUSTOMER, LOAN_APPLICATION are related with ternary relationship.
TERNARY (bid integer, cno integer, lno integer).

Solve the Queries

1. Find the names of the customers for the "Aundh" branch.
2. List the names of the customers who have received loan less than their requirement.
3. Find the maximum loan amount approved.
4. Find out the total loan amount sanctioned by "Deccan "branch.
5. Count the number of loan applications received by "M.G.ROAD" branch.
6. List the names of the customer along with the branch names who have applied for loan in the month of September.

Set C
Student- Teacher database
Consider the following database maintained by a school. The school maintains information about students and the teachers. It also gives information of the subject taught by the teacher.

Following are the tables:
STUDENT (sno integer, s_name char(30), s_class char(10), s_addr char(50))
TEACHER (tno integer, t_name char (20), qualification char(15),experience integer)

The relationship is as follows:
STUDENT-TEACHER: M-M with descriptive attribute SUBJECT.

Solve the queries
1. Find the minimum experienced teacher.
2. Find the number of teachers having qualification “Ph. D.”.
3. List the names of the students to whom “Mr. Patil” is teaching along with the subjects he is teaching to them.
4. Find the subjects taught by each teacher.
5. List the names of the teachers who are teaching to a student named“Suresh”.
6. List the names of all teachers along with the total number of students they are teaching.

Assignment Evaluation

0: Not Done [ ]
1: Incomplete [ ]
2:Late Complete[]
3: Needs Improvement [ ]
4: Complete [ ]
5: Well Done [ ]

Signature of Instructor

Assignment No.13
Case studies

Set A

Business trip database

Consider the business trip database that keeps track of the business trips of salesman in an office.

Following are the tables:

SALESMAN (sno integer, s_name char (30), start_year year, deptno varchar (10))
TRIP (tno integer, from_city char (20), to_city char (20), departure_date date, return date)
DEPT (deptno varchar (10), dept_name char(20)) ,expense (eid integer, amount money)
The relationship is as follows
DEPT-SALESMAN 1 TO M
SALESMAN - TRIP 1 TO M
TRIP - EXPENSE 1 TO 1

Execute the following queries

1. Increase the expenses of all the trips by Rs. 5000.
2. Give the details for trips that exceed Rs. 10,000 in expenses.
3. List the salesman numbers and names of the salesmen who made trips to Calcutta.
4. Delete all the trips made by department "computer" having expenses more than Rs.15000.
5. Find the departments from which the salesmen have done highest number of trips.
6. Find the total expenses incurred by the salesman "Mr. Patil".

Set B

Warehouse Database

CITIES (city char (20), state char(20))
WAREHOUSES (wid integer, wname char (30), location char(20))
STORES (sid integer,store_name char(20), location_city char(20))
ITEMS (itemno integer, description text, weight decimal(5,2), cost decimal(5,2))
CUSTOMER(cno integer, cname char(50),addr varchar(50), c_city char(20))
ORDERS(o_no int, o_date date)

The relationship is as follows

CITIES-WAREHOUSES 1 TO M
WAREHOUSES - STORES 1 TO M
CUSTOMER – ORDERS 1 TO M
ITEMS – ORDERS M TO M relationship with descriptive attribute ordered_quantity
STORES-ITEMS M TO M RELATION with descriptive attribute quantity

Solve the following queries.

1. Find the item that has minimum weight.
2. Find the different warehouses in "Pune".
3. Find the details of items ordered by a customer "Mr. Patil".
4. Find a Warehouse which has maximum stores.
5. Find an item which is ordered for minimum number of times.
6. Find the details orders given by each customer.

Set C

movie database

movies(m_name, release_year, budget)

actor(a_name, role, charges, a_address)

producer(producer_id, name, p_address)

each actor has acted in one or more movies. each producer has produced many movies and each movie can be produced by more than one producers. each movie has one or more actors acting in it, in different roles.

create the relations accordingly, so that the relationship is handled properly and the relations are in normalized form(3nf).

insert sufficient number of appropriate records.

solve the queries:

1. list the names of actors who have acted in at least one movie, in which '________' has acted.
2. list the names of the movies with the highest budget.
3. list the names of actors who have acted in the maximum number of movies.
4. list the names of movies, produced by more than one producer.
5. list the names of actors who are given with the maximum charges for their movie.
6. list the names of producers who produce the same movie as '_________'.
7. list the names of actors who do not live in__________or__________city.

Assignment Evaluation

0: Not Done [ ]   1: Incomplete [ ]   2:Late Complete[]

3: Needs Improvement [ ]   4: Complete [ ]   5: Well Done [ ]

Signature of Instructor