TYPES OF SQL STATEMENTS

Data Definition language (DDL)	Data manipulation language (DML)	Data control language (DCL)	Transition control language (TCL)
CREATE	SELECT	GRANT	COMMIT
ALTER	INSERT	REVOKE	ROLL BACK
DROP	UPDATE	ANALYZE	SAVE POINT
TRUNCATE	DECETE	AUDIT	
RENAME	MERGE	COMMENT	
	CALL		
	LOCK TABLE		
	EXPLAIN PLAN		

DATA DEFINITION LANGUAGE

Statement	Description
CREATE	Create new database/table
ALTER	Modifies the database/table
DROP	deletes database/table
TRUNCATE	Remove all table records including allocated table spects
RENAME	Rename the database/table

Transition control language

Statement	Description
SAVEPOINT	Create savepoint for later use
ROLL BACK	Restore database to orginal form since last comm
COMMIT	Permanent work save into database

Date 6/4/18

Expt. No. 1 INTRODUCTION AND CREATION OF TABLE Page No. 1

What is SQL?

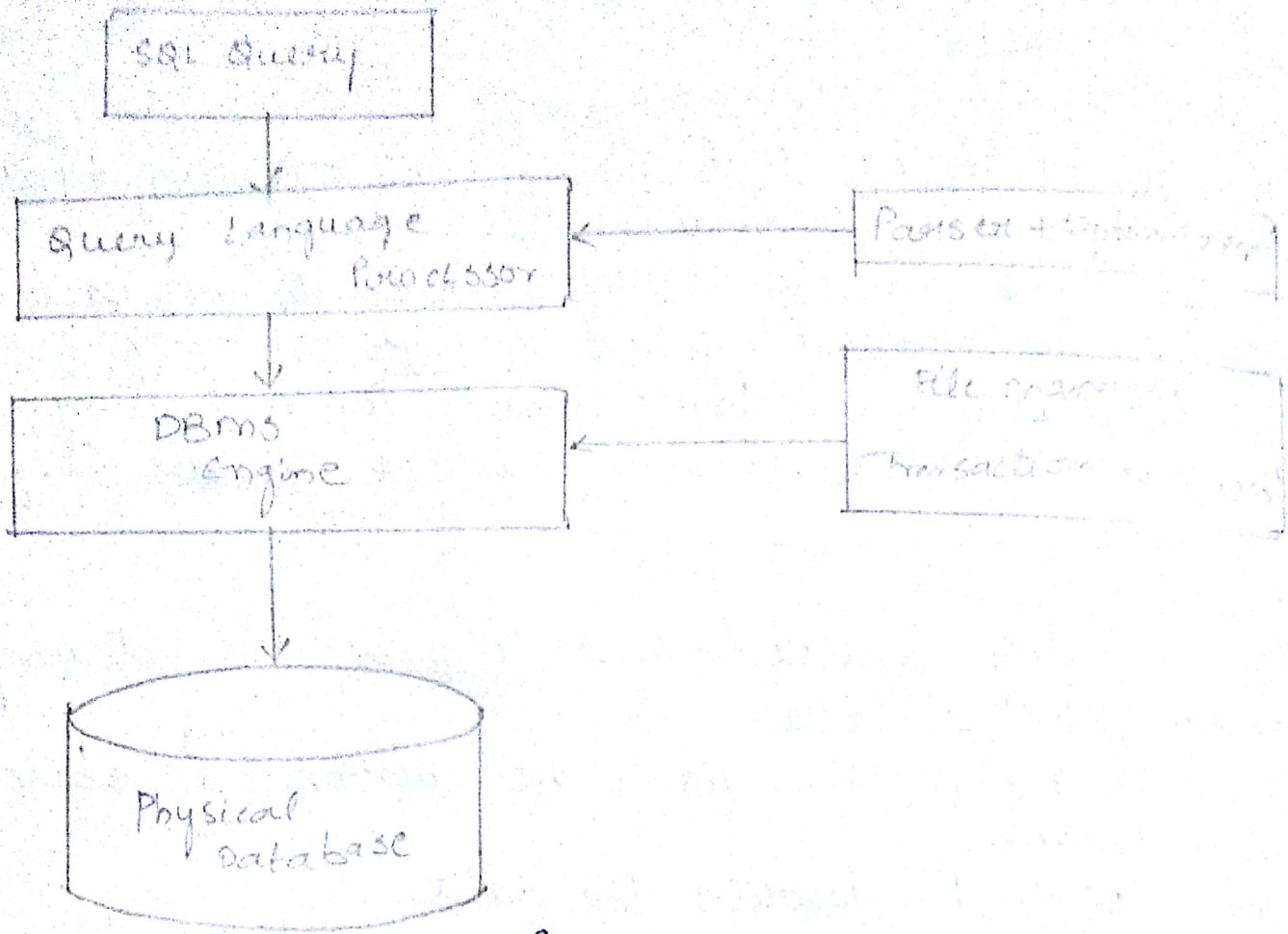
SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database.

SQL is the standard language for Relational Database Systems. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

Why SQL?

SQL is widely popular because it offers the following advantages:-

- Allows users to access data in the relational database management systems.
- Allows users to describe the data
- Allows users to define the data in a database and manipulate that data
- Allows to embed within other languages using SQL, modules libraries & pre-compilers
- Allows users to create and drop databases and tables.
- Allows users to create views, stored procedure, functions in a database
- Allows users to set permissions on tables, procedures and views.



Data Manipulation language

Statement	Description
SELECT	Retrieve data from table
INSERT	Insert data into table
UPDATE	Updating existing data with new data
DELETE	Delete record rows of table
MERGE	Insert new statements or update exiting records

Data control language

Statement	Description
GRANT	Gives priviliges to user for accessing data
REVOKE	Takebacks for given privilage
ANALYZE	Analyze Statements to collect statistical info
AUDIT	Tracks the occurence of specific SQL statement
COMMENT	write comment on table

SQL commands:

SQL commands are instructions coded into SQL statements, which are used to communicate with database to perform specific tasks, works, functions and queries with data.

SQL commands can not only be used for searching the database but also to perform various other functions like creating tables, adding data to tables, modifying data, dropping tables and setting permissions for users.

SQL commands are grouped into four main categories depending on their functionality.

(1) Data Definition Language (DDL): These are SQL commands used for creating, modifying and dropping the structure of database objects. The commands are CREATE, ALTER, RENAME, TRUNCATE.

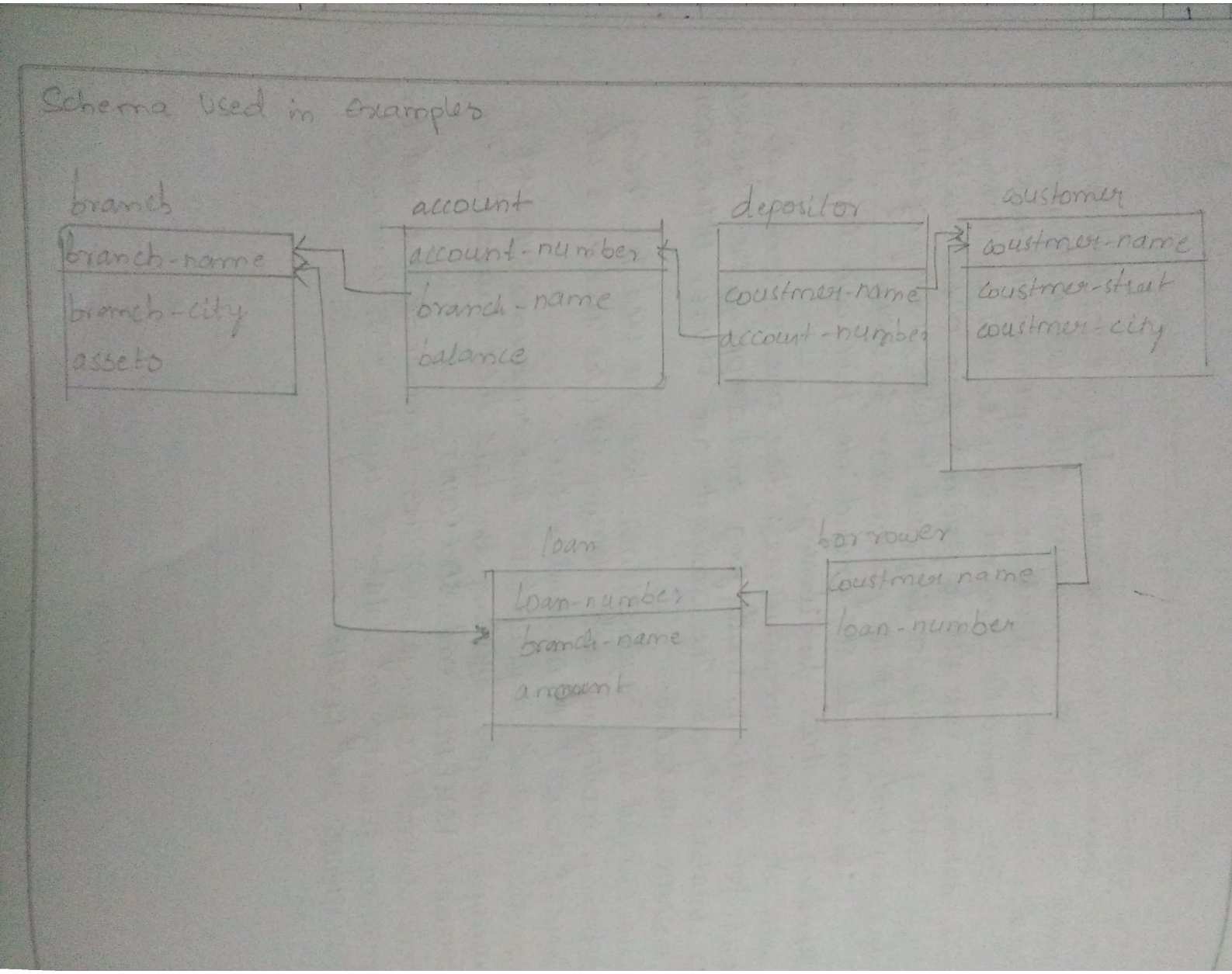
(2) Data Manipulation Language (DML): These are used for storing, retrieving, modifying and deleting data. The commands are SELECT, INSERT, UPDATE and DELETE.

(3) Transition control Language (TCL): These commands are used for managing changes affecting the data. These commands are COMMIT, ROLLBACK and SAVEPOINT

(4) Data Control Language (DCL): These commands are used for providing security to database objects. These commands are GRANT and REVOKE

Teacher's Signature _____

Schema Used in Examples



Data Definition Language (DDL)

* Allows the specification of not only a set of relations but also information about each relation, including:
  * The schema for each relation.
  * The domain of values associated with each attribute.
  * Integrity constraints
  * The set of indices to be maintained for each relations.
  * Security and authorization information for each rela
  * The physical storage structure of each relation on disk.

Domain Types in SQL

* char(n): Fixed length char string, with user-specified length n
* varchar(n): Variable length character strings, with user-specified maximum length n.
* int: Integer
* smallint: Small integer
* numeric(p,d): Fixed point number, with user-specified precis of p digits with n digits to the right of decimal point.
* real, double precision: Floating point and double-precision floating point numbers, with machine-dependent precision
* float(n): Floating point number, with user-specified precisio of at least n digits.

Null values are allowed in all the domain types. Declaring an attribute to be not null prohibits null values for that attribute.

* create domain: construct in SQL-92 creates user-defined domain types.

```
create domain person-name char(20) not null
```

Teacher's Signature

Date/time Types in SQL

date. Dates, containing a (4 digit) year, month and date
eg date '2001-7-27'

time: Time of day, in hours, minutes and seconds
eg time '09:00:30' time: '09:00:30:75'

timestamp: date plus time of day
eg timestamp '2001-7-27 09:00:30.75'

Create Table Construct

An SQL relation is defined using the create table command
create table r(A1 D1, A2 D2 ... An Dn),
(integrity-constraint1),
...,
(integrity-constraint k))

- r is the name of the relation
- each A is an attribute name in the schema of relation r
- D is the data type of value in the domain of attribute A

eg:

```
create table branch (branch-name char(15) not null,
                     branch-city char(30),
                     assets integer)
```

Integrity constraints in Create table

- not null
- primary key (A1, ... An)
- check (P), P is predicate

Teacher's Signature

eg: Declare branch-names as the primary key for branch and ensure that the values of assets are non-negative.

- create table branch
  (branch-name char(15),
  branch-city char(30)
  assets integer,
  primary key(branch-name),
  check(assets >=0))
- primary key: declaration on an attribute automatically ensures not null.

Drop and Alter Table constructs:

The drop table command deletes all information about the dropped relation from the database.

The alter table command is used to add attributes to an existing relation alter table r add A D
where A → name of attribute
D → Domain of A

- All tuples in the relation are assigned null as the value for the new attribute.

The alter table command can also be used to drop attributes of a relation alter table r drop A

SQL NOT NULL Constraint

- By default, a column can hold NULL values.
- The NOT NULL constraint enforces a column to NOT accept NULL values.

```
CREATE TABLE Persons (ID int not null, LastName varchar(255)
NOT NULL, FirstName varchar(255) NOTNULL, Age int);
```

SQL UNIQUE Constraint

- The UNIQUE constraint ensures that all values in a column are different.
- Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.
- A primary key constraint automatically has a unique constraint.
- However, you can have many UNIQUE constraints per table but only one PRIMARY KEY constraint per table.

eg:-

```
create table persons( ID int not null unique, LastName varcha
not null );
```

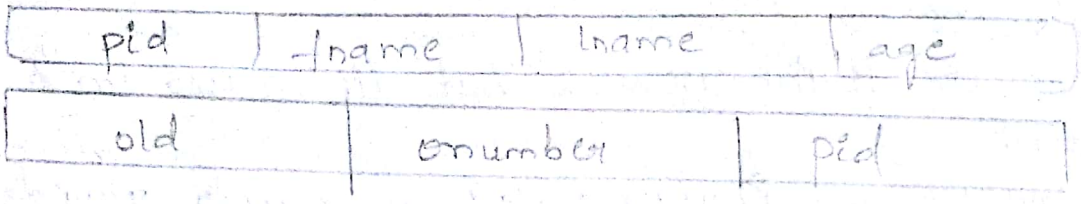
SQL PRIMARY KEY Constraint

- The PRIMARY KEY constraint uniquely identifies each record in a database table
- They should contain UNIQUE values and cannot contain NULL values.
- A table can have only one primary key, which may contain by single or multiple fields.

Teacher's Signature

SQL Foreign key Constraint

- A Foreign key is a key used to link two tables together
- A Foreign key is a field in one table that refers to the primary key in another table
- The table containing the foreign key is called the child table, and the table containing the candidate key is called the referenced or parent table.



create table orders(orderID int NOT NULL PRIMARY key, orderNumber int NOT NULL, PersonID int FOREIGN KEY REFERENCES Persons(PersonID));

SQL CHECK Constraint

The check constraint is used to limit the value range that can be placed in a column.

If you define a CHECK constraint on a single column it allows only certain values for this column,

CREATE TABLE Persons (
ID int NOT NULL, Age int CHECK (Age >=18));

SQL DEFAULT Constraint

The DEFAULT constraint is used to provide a default value for a column.

The default value will be added to all new records IF one other value is specified

```
CREATE TABLE persons( city varchar(255) DEFAULT 'Sa
                                                    );
```

The DEFAULT constraint can also be used to insert system values, by using functions like GETDATE():

```
CREATE TABLE Orders( OrderDate date DEFAULT GETD
                                                    );
```

SQL Working with Dates

DATE format YYYY-MM-DD
DATETIME format YYYY-MM-DD HH:MI:SS
SMALLDATETIME format YYYY-MM-DD HH:MI:SS
TIMESTAMP format: a unique number

Basic Structure

* A typical SQL query
  select A1, A2...An from r1, r2...rm where P
where,
  A is represent attributes
  r is represent relations
  P is predicate
* The result of an SQL query is a relation

The Select Clause

* The select clause list the attributes desired in the result of a query
eg:- find the names of all branches in the loan relation.
  select branch-name from loan
* SQL does not permit the '-' character in names
* SQL allows duplicates in relations as well as in query results.
* To force the elimination of duplicates, insert the keyword distinct after select
* Find the names of all branches in the loan relations, and remove duplicates.
select distinct branch-name
  from loan
The keyword all specifies that duplicates not be removed
  select all branch-name from loan
* An asterisk in the select clause denotes "all attributes"
  select * from loan
* The select clause can contain arithmetic expressions involving the operation +, -, * and /

Teacher's Signature _____

The query:

Select loan-number, branch-name, amount * 100 from loan

would return a relation which is the same as the loan relations, except that the attribute amounts multiplied by 100

The where clause

The where clause specifies conditions that the result must satisfy

To find all loan

Select loan-number from loan where branch-name = 'Perryridge' and amount > 1200

Comparison results can be combined using the logical connectives and, or and not

Comparisons can be applied to results of arithmetic expressions

SQL includes a between comparison operator

eg Find the loan number of those loans with loan amounts between $90,000 and $100,000 (that is, ≥ $90,000 and ≤ $100,000)

Select loan-number from loan where amount between 90000 and 100000

The from clause lists the relations involved in the query

Find the cartesian product borrower x loan.

Select * from borrower, loan

Find the name, loan number and loan amount of all customers having a loan at the perryridge branch

Select customer-name, borrower.loan-number, amount from borrower, loan where borrower.loan-number = loan.loan-number and branch-name = 'Perryridge'

The Rename Operation

The SQL allows renaming relations and attributes using the as clause

old-name as new-name

Tuple variables

They are defined in the from clause via the use of the as clause

Find the coustmer names and their loan numbers for all coustmers having a loan at some branch

Select customer-name, T.loan-number, S.amount from borrower as T, loan as S where T.loan-number = S.loan-number.

String Operations

- SQL includes a string-matching operator for comparisons on character strings. patterns are described using two special characters:
  - percent (%). The % character matches any substring
  - underscore (_). The _ character matches any character
- Find the names of all customers whose street includes the substring "Main":

```
select customer-name from customer where customer_street
  like '%Main%'
```

- Match the name "Main%" like 'Main\%' escape '\'
- SQL supports a variety of string operations such as
  - concatenation (using "||")
  - converting from upper to lower (and vice versa)
  - finding string length, extracting substrings etc

Ordering the Display of Tuples

- List in alphabetic order the names of all customers having a loan in Perryridge branch

```
select distinct customer-name from borrower, loan
where borrower loan-number = loan.loan-number and
  branch-name = "Perryridge" order by customer_name
```

Expt. No.

We may specify desc for descending order ; or asc for ascending order, for each attribute ; ascending order is the default

eg:- order by customer-name desc

Set Operations

union, intersect and except operate on relations and correspond to the relational algebra operations

Each of the above operations automatically eliminates duplicates; to retain all duplicates use the corresponding multiset versions union all, intersect all and except all

Suppose a tuple occurs m times in r and n times in s then, it occurs:

m + n times in r union all s

min(m, n) times in r intersect all s

max(0, m-n) times in r except all s

Aggregate Functions

These functions operate on the multiset of values of a column of a relation and return a value

avg: average value

min: minimum value

max: maximum value

sum: sum of values

count: number of values

eg Find the average account balance at the Perryridge branch

Select avg (balance) from account where branch-name = 'Perryridge'

Aggregate Functions - Group By

Find the number of depositors for each branch

Select branch-name, count (distinct customer-name) from depositor, account where depositor.account-number = account.account-number group by branch-name

Attributes in select clause outside of aggregate functions must appear in group by list

Aggregate functions - Having clause

Find the names of all branches where the average account balance is more than $1,200

Select branchname, avg(balance) from account group by branch-name having avg(balance) > 1200

NULL values:

* It is possible for tuples to have a null value, denote by null, for some of their attributes
* null signifies an unknown value or that a value d not exist
* The predicate is null can be used to check for nul values

eg Find all loan number which appear in the l relation with null values for amount

```
select loan-number
from loan
where amount is null
```

Null values and three valued logic

Any comparison with null returns unknown
eg 5<null or null<>null or null=null

Three-valued logic using the truth value unknown
OR: (unknown or true) = true, (unknown or false) = unkn
(unknown or unknown) = unknown

AND: (true and unknown) = unknown, (false and unknown false, (unknown and unknown) = unknown

NOT: (not unknown) = unknown

"P is unknown" evaluates to true if predicate P evalua to unknown

Result of where clause predicate is treated as false if it evaluates to unknown.

NULL values and Aggregates

Total all loan amounts

```
Select sum(amount)
from loan
```

- above statement ignores null amounts
- result is null if there is no null amount
- All aggregate operations except count(*) ignore tuples with null values on the aggregated attributes

COMPANY DATABASE AND QUERIES RELATED TO DATABASE

```
drop table employee11;
create table employee11 (
        Fname  varchar(20) not null,
        minit  char(1),
        Lname  varchar(20) not null,
        Ssn  number(9) primary key,
        Bdate  date  not null,
        Address  varchar(30) not null,
        Sex char(1)
        Salary  number(10),
        Super_ssn  number(9),
        FOREIGN KEY (Super_ssn) references employee11(ssn)
);

drop table department;
create table department (
    Dname  varchar(20) not null,
    Dnumber  number(10) unique,
    mgr_ssn  number(10) unique,
    FOREIGN KEY (mgr_ssn) references employee(ssn),
    mgr_start_date  DATE
);

alter table employee11 add Dno int constraint FK_Dno1
    references department(Dnumber);
```

employee1)

Fname	Minit	lname	SSN	Bdate	Address	sex	Salary	Supervisor_no
John	B	Smith	123456789	1965-01-09	731 Forden Houston, TX	M	30000	33344555 5
Alicia	J	Zelaya	999887777	1968-01-19	3321 castle Spring, TX	F	25000	98765432 4

department

Dname	Dnumber	Mgr_SSN	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1961-06-09

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

```
drop table project;
create table project (
        Pname varchar(20) unique,
        Pnumber number(5) primary key,
        Plocation varchar(20),
        Dno number(10),
        FOREIGN KEY (Dno) references department(Dnumber
);

drop table DEPT_LOCATION;
create table DEPT_LOCATION(
        Dnumber number(10),
        FOREIGN KEY(Dnumber) references department(D
        Dlocation varchar(20),
        Primary key (Dnumber, Dlocation)
);

drop table DEPENDENT;
create table DEPENDENT (
        Essn number(9),
        FOREIGN KEY(Essn) references employee11(SSn),
        Dependent_name varchar(20),
        Sex char(1),
        Bdate date)
        Relationship varchar(20),
        primary key(Essn, Dependent_name)
);
```

Project

Pname	Pnumber	Plocation	Dno
Product X	1	Bellaire	5
Product Y	2	Sugarland	5
Computerization	10	Stafford	4

Dependent

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
123456789	Michael	M	1988-01-04	Son

Works_on

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0

```
drop table works_on;
create table works_on(
        Essno number(10),
        Pno number(5),
        FOREIGN KEY (Essn) references employee11(SSn),
        FOREIGN KEY (Pno) references Project(Pnumber),
        Hours number(2),
        Primary key (Essn, Pno)
);
```

Insertions:-

```
insert into employee11(Fname, minit, lname, SSn, Bdate,
Address, Sex, salary, Super_ssn, Dno) values ('John', 'B',
'Smith', 123456789, '09-Jan-1965', '731, Fordren, Houston,
'M', 30000, 333445555);
insert into employee11 values ('Alicia', 'J', 'Zelaya', 99988
'19-Jan-1968', '3321 Castle, Spring, TX', 'F', 25000, 987654

insert into DEPT_LOCATIONS values (1, 'Houston');
insert into DEPT_LOCATION values (4, 'Stafford');

insert into project values ('Product X', 1, 'Belaire', 5);
insert into project values ('computerization', 10, 'Stafford', 4)

insert into department values ('Research', 5, 333445555, '22
198

insert into department values ('Administration', 4, 987654321,
'01-Jan-1995');
```

Teacher's Signature _____

insert into works_on values (123456789, 1, 32);
insert into works_on (Essn, pno, Hours) values (123456789, 2, 8);

insert into DEPENDENT (Essn, Dependent_name, sex, Bdate, Relation
values (333445555, 'Alice', 'F', '04-may-1985', 'Daughter');

insert into DEPENDENT (Essn, Dependent_name, sex, Bdate, Relatio
values (333445555, 'Theodore', 'M', '03-May-1985', 'Spouce');

1)

Bdate	Address
1965-01-09	731 Fonden, Houston, TX

2)

Fname	Lname	Address
John	Smith	731 Fondren, Houston, TX
Franklin	Wong	638 Voss, Houston, TX
Ramesh	Narayan	915 Fire Oak, Humble, TX
Ahmad	Jabbar	980 Dallas, Houston, TX

3)

pnumber	dnumber	dname	address	bdate
10	4	Walker	291, Berry, TX	20-06-1941
30	4	Walker	291, Berry, TX	20-06-1941

Fname	address
Franklin	638 Voss, Houston, TX
John	731 Founda!, Houston, TX

Querys

1. Retrieve the birthdate and address of the employee(s) whose name is John.B.Smith.

Select bdate, address from employee1 where fname='John' and minit='B' and lname='Smith';

2. Retrieve the name and address of the employees who work for research department

Select fname, lname, address from employee, department where dnumber=dno and dname='Research';

3. For every project located in Stafford list the project number, the controlling department number and the dept managers lastname, address and birthdate.

Select pnumber, dnum, lname, address bdate from project, department, employee11 where dnum=dnumber, mgrssn=SSn and plocation='Stafford';

Retrieve the firstname, address of managers working for research department

Select fname, address from employee11 department where mgr.ssn=SSn and dname='research';

Teacher's Signature _____

5)

Fname	lname	fname	lname
John	Smith	Franklin	wong
Ramesh	Narayan	Franklin	wong

6)

SSN	dname
123456789	Research
999887777	Administration

7)

fname	minit	lname	ssn	bdate	address
John	B	Smith	444444	55-01-09	
Franklin	+	wong	222222	45-12-08	

8)

fname	lname	ssn	dno
~~Ramesh~~	Narayan	666644444	5
Alicia	Zelaya	999887777	4
Jennifer	wallace	987654321	4
Ahmad	jabbar	987987987	4

pt. No.

For each employee, retrieve the employee first and last name and first and last name of his or her immediate supervisor

Select e1.fname, e1.lname, e2.fname, e2.lname from employee e1, employee e2 where e1.sn = e2.superssn

Select all employee SSNs and all combinations of employee ssn and department dname in the database

Select ssn from employee
Union
Select ssn, dname from employee, department where
dno=dnumber

Select all attributes of employees who are working for department (Administration) 5

Select * from employee where dno='5'

Select all attributes of employees who are working for department administration

Select * from employee, department where dname = 'Admins
and employee.dno = department.dnumber

10)

Salary
30000
40000
25000
43000
38000
25000
25000
55000

11)

distinct Salary
30000
40000
25000
43000
38000
55000

12)

13)

Fname	Lname	address
Joyce	English	Houston, TX
Alimad	Jabber	Houston, TX
James	Borg	Houston, TX

9) Select all attributes from employee and department

Select * from employee union select * from department;

10) Retrieve the salary of every employee

Select salary from employee;

11) Retrieve distinct salary of every employee

Select distinct(salary) from employee;

12) Make a list of all projects numbers for projects that involved an employee whose last name is 'smith' either as a worker or as a manager of department that controls the project

Select distinct pno from project, department, employee where dnum = dnumber and mgrssn = ssn and lname = 'Smith's

Union

Select distinct pno from project, dept, employee where pnum = pno and essn = ssn and dno = dnumber and lname = 'smith';

13) Retrieve all employees whose address is in the Houston

Select * from employee where address like '%Houston Texas%';

Teacher's Signature _____

14)

Fname	Lname	bdate
Franklin	wong	1955-12-08

15)

Fname	Lname	product-name	Sal
John	Smith	product X	33000
Franklin	Wong	product X	41400

16)

pname	lname	Salary
John	Smith	30000
Franklyn	Wong	40000
Ramesh	Narayan	38000

17)

dname	lname	fname	pname
Research	Smith	John	product X
Research	wong	Franklyn	product Y
administration	zelaya	Alicia	Computerization

Find as employee who were born during the year '1950'

Select * from employee where bdate like
'- - - - - - - - 1950';

Show the resulting salaries if every employee working on the product 'X', project is given a 10% raise

Select *
Salary + Salary * 0.1 as Sal from employee e,
project p, works_on w where
e.ssn = w.essn
w.pno = p.pnumber
p.pname = 'productX';

Retrieve all employees in department and 5 whose salary is btw 30,000 and 40000

Select * from employee where dno=5 and salary between 30000 and 40000;

Retrieve a list of employees and the projects they are working ordered by department and within each department, ordered alphabetically by lname and fname

Select * from employee e, projects p, department d, workson w where
e.ssn = w.essn, w.pno = p.pnumber,
d.dnumber = e.dno and order by d.name,

Teacher's Signature

e.fname and e.lname;

3) Delete from employee where lname is 'brown'

delete from works_on, dependent, employee where lname='brown' and essn=ssn;

1) Delete from employee where ssn is '123456789'

delete from works_on, dependent, employee where essn='123456789' and essn=ssn;

5) Delete from employee where department number is 5.

delete from works_on, dependent, employee where dno=5 and essn=ssn;

Update in values of project location to Bellaire for dnum=5 and pnumber=10

Update project SET plocation='Bellaire', dnum=5 where pnumber=10;

Update in the table employee, where salary is incremented by 10% and dnumber is 10

Update employee SET salary=salary*0.1 where dnumber

Teacher's Signature _____

23) insert into employee (fname, lname, dno, ssn);
values ('Richard', 'marini', 4, 653298653);

Region_cd	city	salesperson	Sales	gender
N	lucknow	Trivedi, virendra	460	M
N	lucknow	Singhal, aruna	550	F
W	Bombay	kulkarni, shobha	680	F
E	Calcutta	verma, rachit	630	M

city	salesperson	sales
Bombay	kulkarni, shobha	680
Calcutta	Chatterji, Suresh	640

city	gender	average_sales
N		

Date 3/9/18

Expt. No. 4 — Sales Database — Page No. 20

```
create table Sales_Data(Region_cd char(1), city varchar(30), Salesperson
varchar(30), Sales number(30,0), gender char(1));

insert into Sales_Data values ('N', 'lucknow', 'Trivedi, virendra', 460, 'M');
insert into Sales_Data values ('N', 'lucknow', 'singhal, aruna', 550, 'F');
insert into Sales_Data values ('N', 'lucknow', 'khan, arif', 625, 'M');
insert into Sales_Data values ('N', 'lucknow', 'Joshi, varun', 600, 'M');
insert into Sales_Data values ('N', 'lucknow', 'Robinson, danny', 490, 'M');
insert into Sales_Data values ('N', 'lucknow', 'Singh, Reena', 620, 'F');
insert into Sales_Data values ('N', 'kanpur', 'Singh, Rahul', 650, 'M');
```

Queries:

(1) Select * from Sales_Data;

(2) Select city, sales, Salesperson from Sales_Data where sales >= 600;

(3) Retrive city, gender and average sales of salesperson from Sales_Dat

Select city, gender, avg(sales) from Sales_Data group by city, gender;

4) Retrieve city, gender and average sales where Region_cd is 'N'

Select city, gender, avg(sales) as AVG_Sales from Sales_Data where Region_cd='N' group by city, gender;

) Retrieve Region_cd and city where avg sales are greater than 550

Select Region_cd, city from Sales_Data group by city, Region_cd having avg(sales) > 550

Teacher's Signature

) Retrieve city, salesperson where sales is greater than or equal to 600 and order by city and salesperson

Select city, salesperson from SalesData where Sales >= 600 order by city and salesperson;

) Retsieve city and salesperson where sales are greater than or equal to 600 and order by des city and asc of salesperson

Select salesperson, city from Sales_Data where Sales >= 600 order by city desc, Salesperson asc;

) Retrieve city, salesperson where City = 'Lucknow'

Select city, salesperson from Sales_Data where city = 'Lucknow';

Retrieve city, salesperson, sales where sales is greater than or equal to 600.

Select city, salesperson, sales from Sales_Data where sales >= 600

Retrieve city, salesperson, sales where city is Lucknow and sales is greater than or equal to 600

select city, Salesperson, sales from Sales_Data where city = 'Lucknow' and sales >= 600

Teacher's Signature _____

12) Retrieve city, salesperson, sales where city is 'Lucknow' or sales is greater than or equal to 600

```
select city, salesperson, sales from Sales_Data where not (city='lucknow'
, sales>=600);
```

13) Retrieve city where Region_cd='N'

```
Select city from Sales_Data where Region_cd='N';
```

14) Retrieve distinct city where Region_cd='N'.

```
Select distinct (city) from Sales_Data where Region_cd='N';
```

15) Count the number of tuples in the given table

```
Select count(*) from Sales_Data;
```

16) Retrieve the average sales and also increment by 20% where city is Lucknow

```
Select avg(sales), 0.20* avg(sales) as incremented_sales from
Sales_Data where city='lucknow';
```

17) Count the number of tuples in the given table where city='

```
Select count(*) from Sales_Data where city='Bombay';
```

Teacher's Signature

Retrieve salesperson, where salesperson between 'A' and 'N' order by salesperson

Select salesperson from Sales_Data where salesperson between 'A' and 'N' order by salesperson;

Retrieve city, salesperson where sales not between 500 and 600

Select city, salesperson from Sales_Data where not(Sales between 500 and 600);

Retrieve city, salesperson, sales where city in 'Lucknow' and 'Kanpur'

Select city, salesperson, sales ~~where~~ from Sales_Data where city = 'Lucknow' or City = 'Kanpur';

Retrieve city, salesperson, sales where city not in lucknow and kanpur.

Select city, salesperson, sales from Sales_Data where not(city = 'Lucknow' or city = 'Kanpur');

Retrieve salesperson where salesperson like 'Anil';

Select salesperson where salesperson like '%Anil%';

Teacher's Signature _____

Date

Page No. 24

Retrieve salesperson where the salesperson name ends with 'ab'

Select salesperson from Sales_Data where salesperson like '%ab';

Retrieve city, salesperson where city is 'Lucknow' and 'Kanpur' and Sales greater than or equal to 600

Select city, salesperson from Sales_Data where (city = 'Lucknow' city='Kanpur') and sales >=600;

Write a query to extract the average sales across all salespersons

Select avg(sales) from Sales_Data; where :

Write a query to extract all Calcutta based salespersons whose names start with 'T'.

Select salesperson from sales_data where city='calcutta' and salesperson like 'T%';

Write a query to extract max sales in each city with the names of the salesperson

Select max(sales), city from sales_data group by city;

Teacher's Signature