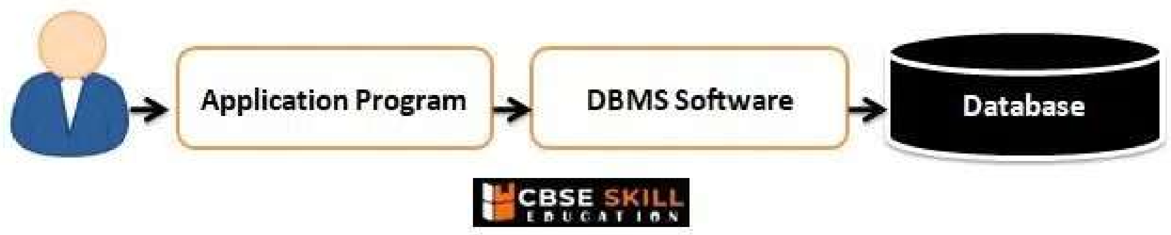
Database Concepts Class 12 Notes

Database Concepts Class 12 Notes

What is Database?

A database is an organized collection of data that has been arranged and is typically kept electronically in a computer system. A database management system often oversees a database (DBMS).



A database has the following properties:

- A database is a representation of some aspect of the real world also called miniworld. Whenever there are changes in this miniworld they are also reflected in the database.
- It is designed, built and populated with data for specific purpose.

- It can be of any size and complexity.
- It can be maintained manually or it may be computerized.

Need for a Database

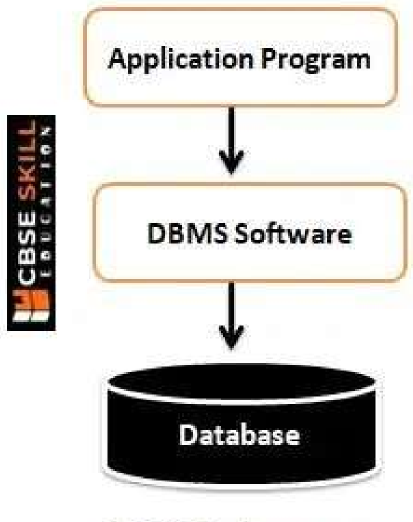
Database management systems enable users to securely, efficiently, and quickly share data throughout an organization. A data management system offers quicker access to more accurate data by quickly responding to database requests.

Database approach –

- **Data Redundancy –** Data redundancy is the storing of the same data across many files. Space would be wasted as a result of this.
- **Data Inconsistency –** If a file is modified, all the files that contain comparable information must also be updated, or the data will become inconsistent.
- **Lack of Data Integration –** Because data files are unique, it is very challenging to obtain information from various files.

Database Management System (DBMS)

Data is stored, retrieved, and analyzed using software called database management systems (DBMS). Users can create, read, update, and remove data in databases using a DBMS, which acts as an interface between them and the databases.



DBMS Environment

The various operations that need to be performed on a database are as follows –

- **Defining the Database –** It involves specifying the data type of data that will be stored in the database and also any constraints on that data.
- **Populating the Database –** It involves storing the data on some storage medium that is controlled by DBMS.
- **Manipulating the Database –** It involves modifying the database, retrieving data or querying the database, generating reports from the database etc.
- **Sharing the Database –** Allow multiple users to access the database at the same time.
- **Protecting the Database –** It enables protection of the database from software/ hardware failures and unauthorized access.
- **Maintaining the Database –** It is easy to adapt to the changing requirements. Some examples of DBMS are – MySQL, Oracle, DB2, IMS, IDS etc.

Characteristics of Database Management Systems

- ***Self-describing Nature of a Database System* –** A database system is said to as self-describing if it has metadata that defines and explains the data and relationships between tables in the database in addition to the database itself.
- ***Insulation Between Programs and Data* –** Programs that access this data don't need to be changed because the description of the data is stored separately in the database management system (DBMS) and any changes to the data's structure are made in the catalogue.
- ***Sharing of Data* –** Multiple users can access the database. Therefore, a DBMS must have concurrency control software to provide concurrent access to the database's data without encountering any consistency issues.

Types of Users of DBMS

Depending on their needs and how they interact with the DBMS, different types of users use the DBMS. Four main categories of users exist –

- **End Users –** those who use the database to perform queries, make changes, and produce reports based on their requirements is a end users.

- **Database Administrator (DBA) –** The DBA is incharge of authorising access, keeping an eye on how it's being used, offering technical support, and acquiring hardware and software resources.
- **Application Programmers –** To communicate with the database, application developers create application programmes. To communicate with the database, these programmes are created using high level languages like SQL.
- **System Analyst –** A system analyst is important to the feasibility, technical, and economic elements of database architecture.

Advantages of using DBMS Approach

Following are the advantages of using a DBMS –

1. **Reduction in Redundancy –** All the data is stored at one place. There is no repetition of the same data. This also reduces the cost of storing data on hard disks or other memory devices.
2. **Improved Consistency –** The chances of data inconsistencies in a database are also reduced as there is a single copy of data that is accessed or updated by all the users.
3. **Improved Availability –** Same information is made available to different users. This helps sharing of information by various users of the database.
4. **Improved Security –** The DBA can protect the database by using passwords and restricting users' database access rights.
5. **User Friendly –** Because of its user-friendly interface, it reduces users' dependence on computer specialists to carry out various data-related actions in a DBMS.

Limitations of using DBMS Approach

1. **High Cost –** The cost of implementing a DBMS system is very high. It is also a very time consuming process.
2. **Security and Recovery Overheads –** Depending on the data stored, unauthorised access to a database can result in a threat to the individual or business. Additionally, regular data backups are necessary to guard against disasters like fires and earthquakes.

Relational Database

A collection of data elements with pre-established relationships between them make up a relational database. These things are arranged in a series of tables with rows and columns. To store data about the things that will be represented in the database, tables are utilised.

<u>In relational model,</u>

1. A row is called a Tuple.
2. A column is called an Attribute.
3. A table is called as a Relation.
4. The data type of values in each column is called the Domain.
5. The number of attributes in a relation is called the Degree of a relation.
6. The number of rows in a relation is called the Cardinality of a relation.

Relational Model Constraints

Constraints are limitations on the values that are stored in a database according to the specifications.

We describe below various types of constraints in Relational model –

- **Domain Constraint –** User-defined columns called domain constraints allow users to enter values in accordance with the data type. Additionally, if it receives an incorrect input, it alerts the user that the column needs to be filled out correctly.
- **Key Constraint –** A primary key constraint is a column or group of columns that shares the same characteristics as a unique constraint. Relationships between tables can be specified using a primary key and foreign key constraints.
- **Null Value Constraint –** A column may by default contain NULL values. A column must not accept NULL values according to the NOT NULL constraint. This forces a field to always have a value, thus you cannot add a value to this field while adding a new record or updating an existing record.
- **Entity Integrity Constraint –** The primary key cannot be null due to the Entity Integrity Constraint. Individual records in a table are identified by a primary key, and if the primary key is null, we are unable to do so. Except for the main key column, any place in the table can have null values.
- **Referential Integrity Constraint –** Foreign key constraints or referential integrity constraints. A logical rule governing the values in one or more

columns in one or more tables is known as a foreign key constraint, also known as a referential constraint or a referential integrity constraint. For instance, a group of tables presents details about the suppliers to a company.

Data types commonly used

Data type	Meaning	Example
CHAR (n)	Fixed length character string. ‘n’ is the number of characters.	CHAR(5):“Ashok” “Vijay”
VARCHAR(n)	Variable length character string. ‘n’ is the maximum number of characters in the string.	VARCHAR(15): “Vijay Kumar” “Ashok Sen”
DATE	Date in the form of YYYY-MM-DD	DATE: ‘2014-03-20’
INTEGER	Integer number	23 56789
DECIMAL (m, d)	Fixed point number m represents the number of significant digits that are stored for values and d represents the number of digits that can be stored following the decimal point. If d is zero or not specified then the value does not contains any decimal part.	DECIMAL(5,2) : 999.99 -567.78 DECIMAL (5) : 23456 99999

Structured Query Language (SQL)

RDBMS data management is done using the SQL language. It is made up of two languages: Data Definition Language (DDL) and Data Manipulation Language (DML), where DDL is a language used to specify the structure and restrictions of data and DML is used to add, alter, and delete data in a database.

Create a Database –

```
CREATE DATABASE School;
```

Create Table Command –

```
CREATE TABLE<table name>
(
<column 1><data type> [constraint] ,
<column 2><data type>[constraint],
<column 3><data type>[constraint]
);
```

Question > Write a Query to Create a new table where the field will be Teacher_ID, First_Name, Last_Name, Gender, Date_of_Birth, Salary, Dept_No.

```
CREATE TABLE Teacher
(
Teacher_ID INTEGER,
First_Name VARCHAR(20),
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2),
Date_of_Birth DATE,
Dept_No INTEGER
);
```

Output –

Teacher_ID	First_Name	Last_Name	Gender	Salary	Date_of_Birth

Create Table using NOT NULL – An attribute value may not be permitted to be NULL.

```
CREATE TABLE TEACHER
(
Teacher_ID INTEGER,
First_NameVARCHAR(20) NOT NULL,
Last_NameVARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2),
Date_of_Birth DATE,
Dept_No INTEGER
);
```

Create Table using DEFAULT – If a user has not entered a value for an attribute, then default value specified while creating the table.

```
CREATE TABLE TEACHER
(
Teacher_ID INTEGER,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER
);
```

Create a Table using CHECK – In order to restrict the values of an attribute within a range, CHECK constraint may be used.

```
CREATE TABLE TEACHER
(
Teacher_ID INTEGER,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER CHECK (Dept_No<=110)
);
```

Create a Table using KEY CONSTRAINT – Primary Key of a table can be specified in two ways. If the primary key of the table consist of a single attribute, then the corresponding attribute can be declared primary key along with its description.

```
CREATE TABLE TEACHER
(
Teacher_ID INTEGER PRIMARY KEY,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER
);
```

Create a Table using REFERENTIAL INTEGRITY CONSTRAINT – This constraint is specified by using the foreign key clause.

```
CREATE TABLE Teacher
(
Teacher_ID INTEGER PRIMARY KEY,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER,
FOREIGN KEY (Dept_No) REFERENCES Department(Dept_ID)
);
```

Naming of Constraint

In the Create Table command, constraints can be named. The benefit is that using the Alter Table command, specified restrictions can be quickly altered or deleted. When naming a constraint, use the keyword CONSTRAINT followed by the constraint's name and its specification.

For example consider the following Create Table command –

```
CREATE TABLE Teacher
(
Teacher_ID INTEGER,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER,
CONSTRAINT TEACHER_PK PRIMARY KEY (Teacher_ID),
CONSTRAINT TEACHER_FK FOREIGN KEY (Dept_No) REFERENCES
Department(Dept_ID) ON DELETE SET NULL ON UPDATE SET NULL
);
```

In the above table, the primary key constraint is named as TEACHER_PK and the foreign key constraint is named as TEACHER_FK.

Drop Table Command

This command is used to delete tables. For example, suppose you want to drop the Teacher table then the command would be:

```
DROP TABLE Teacher CASCADE;
```

Thus Teacher table would be dropped and with the CASCADE option, i.e. all the constraints that refer this table would also be automatically dropped.

However if the requirement is that the table should not be dropped if it is being referenced in some other table then RESTRICT option can be used as shown below:

```
DROP TABLE Teacher RESTRICT;
```

Alter Table Command

Adding a column – Suppose we want to add a column Age in the Teacher table. Following command is used to add the column –

```
ALTER TABLE Teacher ADD Age INTEGER;
```

Dropping a column – A column can be dropped using this command but one must specify the options (RESTRICT or CASCADE) for the drop behavior. RESTRICT would not let the column be dropped if it is being referenced in other tables and CASCADE would drop the constraint associated with this column in this relation as well as all the constraints that refer this column.

```
ALTER TABLE Teacher DROP Dept_No CASCADE;
```

Dropping keys – A foreign key/primary key/key can be dropped by using ALTER TABLE command.

```
ALTER TABLE Teacher DROP FOREIGN KEY TEACHER_FK;
```

Adding a Constraint – If you want to add the foreign key constraint TEACHER_FK back, then the command would be –

```
ALTER TABLE Teacher ADD CONSTRAINT TEACHER_FK FOREIGN KEY (Dept_No)
REFERENCES Department(Dept_ID) ON DELETE SET NULL ON UPDATE SET NULL;
```

Insert Command

This command is used to insert a tuple in a relation. We must specify the name of the relation in which tuple is to be inserted and the values. The values must be in the same order as specified during the Create Table command. For example, consider the following table Teacher:

```
CREATE TABLE Teacher
(
Teacher_ID INTEGER,
First_Name VARCHAR(20) NOT NULL,
Last_Name VARCHAR(20),
Gender CHAR(1),
Salary DECIMAL(10,2) DEFAULT 40000,
Date_of_Birth DATE,
Dept_No INTEGER,
CONSTRAINT TEACHER_PK PRIMARY KEY (Teacher_ID),
);
```

To insert a tuple in the Teacher table INSERT command can be used as shown below:
INSERT INTO Teacher VALUES (101,”Shanaya”, “Batra”, ‘F’, 50000, ‘1984-08-11’, 1);

Update Command

This command is used to update the attribute values of one or more tuples in a table.

```
UPDATE Teacher
SET Salary=55000
WHERE Teacher_ID=101;
```

Delete Command

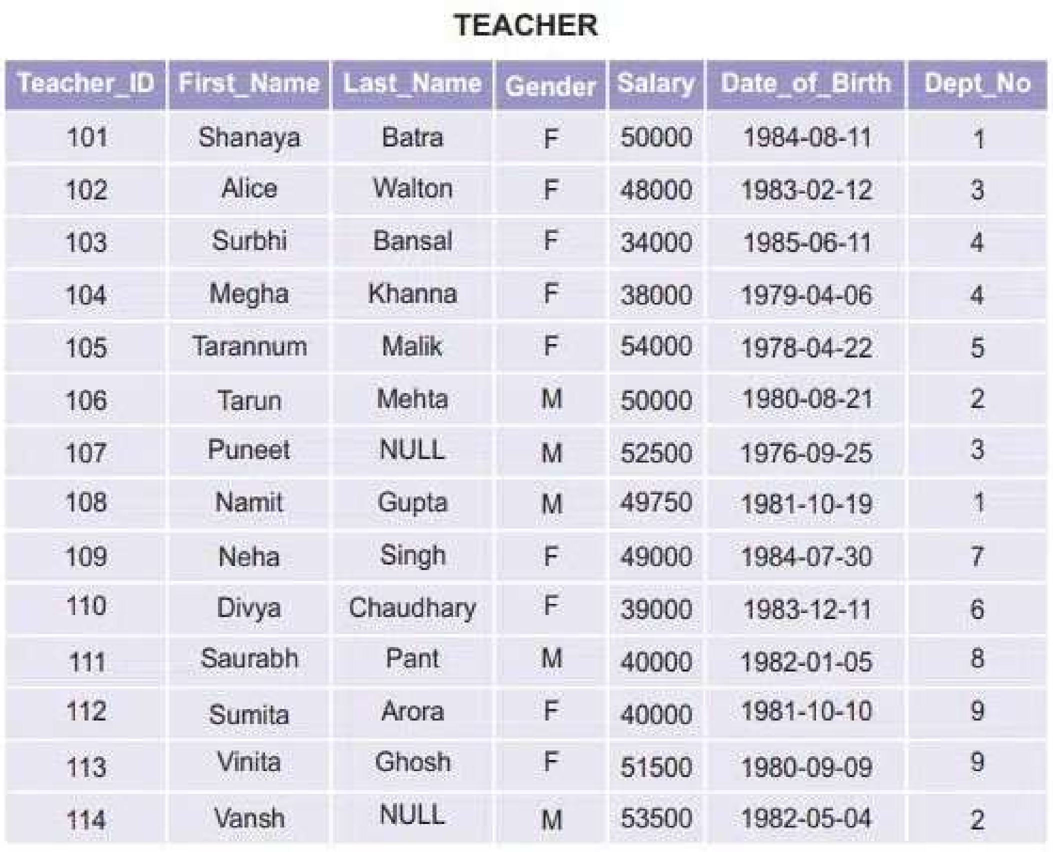
In order to delete one or more tuples, DELETE command is used.

```
DELETE FROM Teacher
WHERE Teacher_ID=101;
```

Select Command

The SELECT Command is used to retrieve information from a database.

```
SELECT <attribute list>
FROM <table list>
WHERE <condition>
```



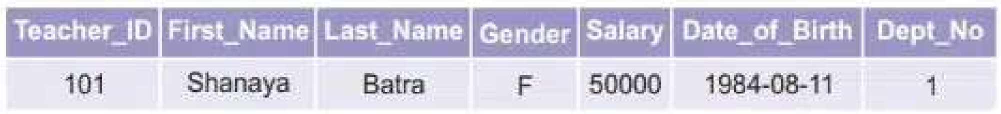
TEACHER

Teacher_ID	First_Name	Last_Name	Gender	Salary	Date_of_Birth	Dept_No
101	Shanaya	Batra	F	50000	1984-08-11	1
102	Alice	Walton	F	48000	1983-02-12	3
103	Surbhi	Bansal	F	34000	1985-06-11	4
104	Megha	Khanna	F	38000	1979-04-06	4
105	Tarannum	Malik	F	54000	1978-04-22	5
106	Tarun	Mehta	M	50000	1980-08-21	2
107	Puneet	NULL	M	52500	1976-09-25	3
108	Namit	Gupta	M	49750	1981-10-19	1
109	Neha	Singh	F	49000	1984-07-30	7
110	Divya	Chaudhary	F	39000	1983-12-11	6
111	Saurabh	Pant	M	40000	1982-01-05	8
112	Sumita	Arora	F	40000	1981-10-10	9
113	Vinita	Ghosh	F	51500	1980-09-09	9
114	Vansh	NULL	M	53500	1982-05-04	2

1. Query – To retrieve all the information about Teacher with ID=101. In this query we have to specify all the attributes in the SELECT clause. An easier way to do this is to use asterisk (*), which means all the attributes.

```
SELECT *
FROM Teacher
WHERE Teacher_ID=101;
```

Output –

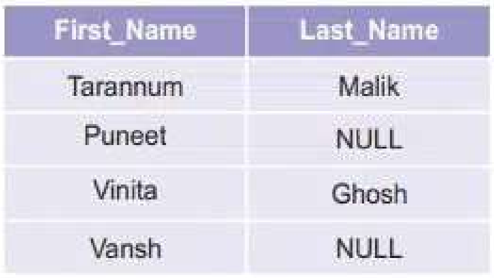


Teacher_ID	First_Name	Last_Name	Gender	Salary	Date_of_Birth	Dept_No
101	Shanaya	Batra	F	50000	1984-08-11	1

2. Query – To find the names of all teachers earning more than 50000.

```
SELECT First_Name,Last_Name
FROM Teacher
WHERE salary > 50000;
```

Output –

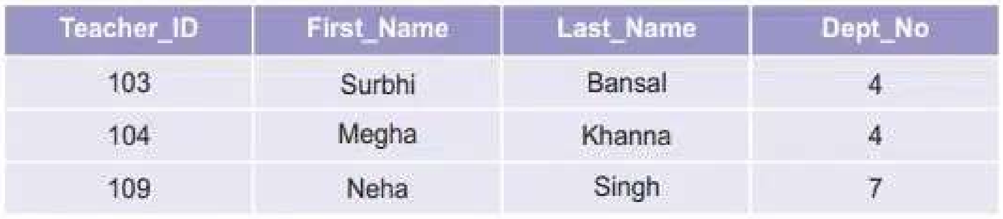


First_Name	Last_Name
Tarannum	Malik
Puneet	NULL
Vinita	Ghosh
Vansh	NULL

3. Query – To display Teacher_ID,First_Name,Last_Name and Dept_No of teachers who belongs to department number 4 or 7.

```
SELECT Teacher_ID,First_Name,Last_Name, Dept_No
FROM Teacher
WHERE Dept_No = 4 OR Dept_No = 7;
```

Output –



Teacher_ID	First_Name	Last_Name	Dept_No
103	Surbhi	Bansal	4
104	Megha	Khanna	4
109	Neha	Singh	7

4. Query – To retrieve names of all the teachers and the names and numbers of their respective departments.
Note that the above query requires two tables – Teacher and Department.
Consider the following query:

```
SELECT First_Name, Last_Name, Dept_ID, Dept_Name
FROM Teacher, Department;
```

5. Query – To retrieve names of all the teachers who belong to Hindi department.

```
SELECT First_Name, Last_Name
FROM Teacher, Department
WHERE Department. Dept_ID=Teacher. Dept_ID AND
Dept_Name="Hindi";
```

First_Name	Last_Name
Surbhi	Bansal
Megha	Khanna

6. Query – To retrieve names of all the teachers starting from letter 'S'.

```
SELECT First_Name
FROM Teacher
WHERE First_Name LIKE "S%";
```

Output –

First_Name
Shanaya
Surbhi
Saurabh
Sumita

7. Query – To retrieve names of all the teachers having 6 characters in the first name and starting with 'S'

```
SELECT First_Name
FROM Teacher
WHERE First_Name LIKE "S _______ ";
```

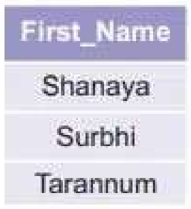
Output –

First_Name
Surbhi
Sumita

8. Query – To retrieve names of all the teachers having at least 6 characters in the first name.

```
SELECT First_Name
FROM Teacher
WHERE First_Name LIKE " _______ %";
```

Output –

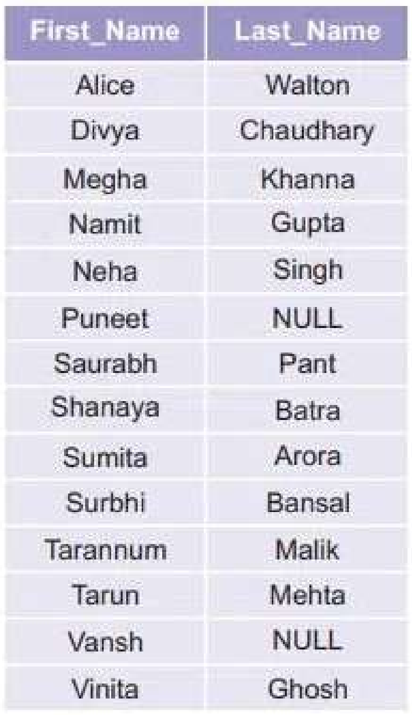


First_Name
Shanaya
Surbhi
Tarannum

9. Query – To list the names of teachers in alphabetical order.

```
SELECT First_Name, Last_Name
FROM Teacher
ORDER BY First_Name, Last_Name;
```

Output –

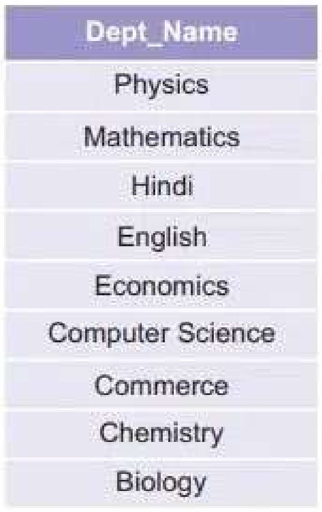


First_Name	Last_Name
Alice	Walton
Divya	Chaudhary
Megha	Khanna
Namit	Gupta
Neha	Singh
Puneet	NULL
Saurabh	Pant
Shanaya	Batra
Sumita	Arora
Surbhi	Bansal
Tarannum	Malik
Tarun	Mehta
Vansh	NULL
Vinita	Ghosh

10. Query – To list the names of all the Departments in the descending order of their names.

```
SELECT Dept_Name
FROM Department
ORDER BY Dept_Name DESC;
```

Output –

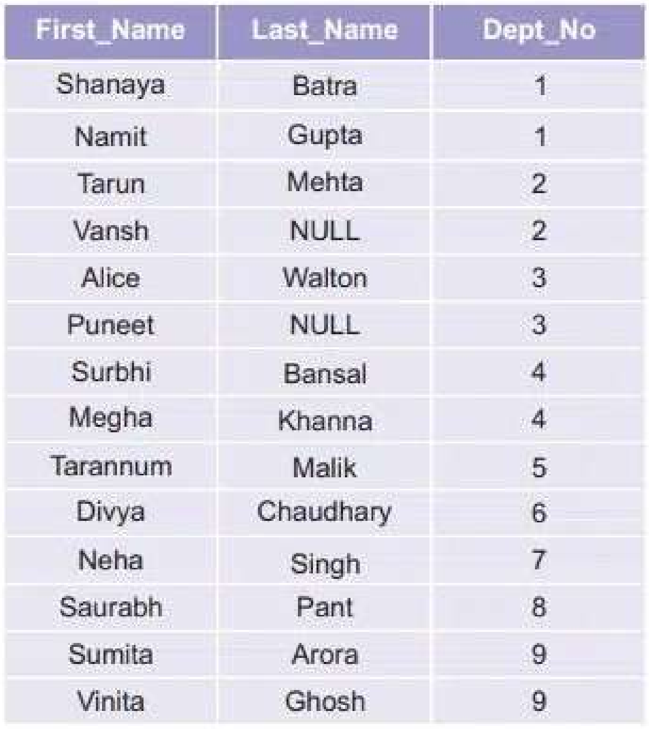


Dept_Name
Physics
Mathematics
Hindi
English
Economics
Computer Science
Commerce
Chemistry
Biology

11. Query – To retrieve the names and department numbers of all the teachers ordered by the Department number and within each department ordered by the names of the teachers in descending order.

```
SELECT First_Name, Last_Name, Dept_No
FROM Teacher
ORDER BY Dept_No ASC, First_Name DESC, Last_Name DESC;
```

Output –

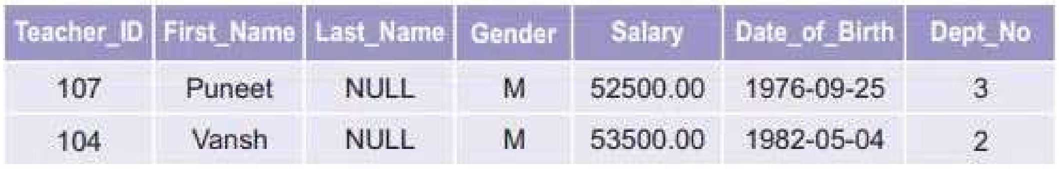


First_Name	Last_Name	Dept_No
Shanaya	Batra	1
Namit	Gupta	1
Tarun	Mehta	2
Vansh	NULL	2
Alice	Walton	3
Puneet	NULL	3
Surbhi	Bansal	4
Megha	Khanna	4
Tarannum	Malik	5
Divya	Chaudhary	6
Neha	Singh	7
Saurabh	Pant	8
Sumita	Arora	9
Vinita	Ghosh	9

12. Query – To retrieve all the details of those employees whose last name is not specified.

```
SELECT *
FROM Teacher
WHERE Last_Name IS NULL;
```

Output –

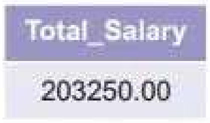


Teacher_ID	First_Name	Last_Name	Gender	Salary	Date_of_Birth	Dept_No
107	Puneet	NULL	M	52500.00	1976-09-25	3
104	Vansh	NULL	M	53500.00	1982-05-04	2

13. Query – To find total salary of all the teachers .

```
SELECT SUM(Salary) AS Total_Salary
FROM Teacher;
```

Output –

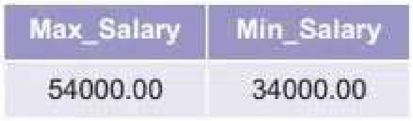


Total_Salary
203250.00

13. Query – To find the maximum and minimum salary.

```
SELECT MAX(Salary) AS Max_Salary, MIN(Salary) AS
Min_Salary
FROM Teacher;
```

Output –

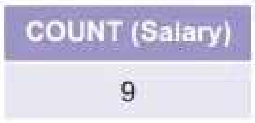


Max_Salary	Min_Salary
54000.00	34000.00

14. Query – To count the number of teachers earning more than Rs 40000.

```
SELECT COUNT(Salary)
FROM Teacher
WHERE Salary > 40000;
```

Output –

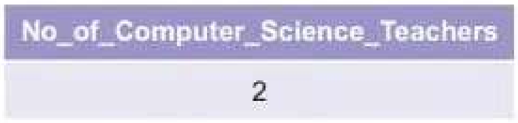


COUNT (Salary)
9

15. Query – To retrieve the number of teachers in "Computer Science" Department.

```
SELECT COUNT(*) AS No_of_Computer_Science_Teachers
FROM Department, Teacher
WHERE Dept_Name = "Computer Science"AND Dept_No=Dept_ID;
```

Output –



No_of_Computer_Science_Teachers
2