

# Database Systems

## Laboratory 1

### Introduction To SQL

RDBMS

Client/Server  
Database

SQL

SQL Statements

SQL\*PLUS

Naming Conventions

Data Types

Table Creation

Viewing Table  
Structure

Record Insertion

Inserting Time

Querying Data

Chittaranjan Pradhan  
School of Computer Engineering,  
KIIT University

# Introduction To SQL

- 1 RDBMS
- 2 Client/Server Database
- 3 SQL
  - SQL Statements
  - SQL\*PLUS
  - Naming Conventions
- 4 Data Types
- 5 Table Creation
- 6 Viewing Table Structure
- 7 Record Insertion
  - Inserting Time
- 8 Querying Data

RDBMS

Client/Server  
Database

SQL

SQL Statements  
SQL\*PLUS  
Naming Conventions

Data Types

Table Creation

Viewing Table  
Structure

Record Insertion  
Inserting Time

Querying Data

## RDBMS

[Client/Server Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

- Relational database is a collection of related information that has been organized into tables. Each table contains rows and columns
- The tables are stored in the database in structures known as **schemas**
- Each row is called an **entity**, thus table as **entity set**
- Row is known as **Tuple**
- Columns are the properties called **Attributes**
- The facts describing an entity are known as **data**
- For an attribute, the set of permitted values is called **domain** of that attribute

# Employee Table

## RDBMS

Client/Server  
Database

## SQL

SQL Statements  
SQL\*PLUS  
Naming Conventions

Data Types

Table Creation

Viewing Table  
Structure

Record Insertion

Inserting Time

Querying Data

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	09-NOV-81	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	23-SEP-81	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

# Examples of RDBMS

- Oracle
- DB2
- Microsoft SQL-Server
- MySQL
- Microsoft Access
- Apache Derby
- Visual FoxPro
- OpenBase
- PostgreSQL
- SQLite
- Vertica
- IBM Informix
- Ingres
- IBM Lotus
- SQL Anywhere

## RDBMS

Client/Server  
Database

### SQL

SQL Statements  
SQL\*PLUS  
Naming Conventions

### Data Types

### Table Creation

Viewing Table  
Structure

### Record Insertion

Inserting Time

### Querying Data

- Client/Server databases run the DBMS as a process on the server and run a client database application on each client
- The client application sends a request for data over the network to the server
- When the server receives the client request, the DBMS retrieves data from the database, performs the required processing on the data, and sends only the requested data back to the client over the network

- SQL( Structured Query Language) is the standard language for relational database
- IBM developed the original version of SQL in early 1970s, with name **Sequel**
- In 1986, the ANSI & ISO published the SQL standard, SQL-86
- ANSI published extended standard, SQL-89 in 1989
- Next versions are SQL-92, SQL-1999, SQL-2003, SQL-2006, SQL-2008, SQL-2011
- SQL is the ideal database language to
  - create database and table structures
  - perform basic data management operation
  - perform complex queries to transform data into useful information

## Data Definition Language(DDL)

Defines the data structure that make up a database

- **CREATE** statement
- **ALTER** statement
- **DROP** statement
- **RENAME** statement
- **TRUNCATE** statement

## Data Manipulation Language(DML)

Modifies the contents of tables

- **INSERT** statement
- **UPDATE** statement
- **DELETE** statement

[RDBMS](#)[Client/Server  
Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table  
Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)



## Query Statement

Retrieves data from the database

- **SELECT** statement

[RDBMS](#)

[Client/Server  
Database](#)

[SQL](#)

[SQL Statements](#)

[SQL\\*PLUS](#)

[Naming Conventions](#)

[Data Types](#)

[Table Creation](#)

[Viewing Table  
Structure](#)

[Record Insertion](#)

[Inserting Time](#)

[Querying Data](#)

## Transaction Control Language(TCL)

Permanently records the changes made to the rows stored in a table or undoes those changes affected by DML statements

- **COMMIT** statement
- **ROLLBACK** statement
- **SAVEPOINT** statement

## Data Control Language(DCL)

Gives and removes permissions on database structure

- **GRANT** statement
- **REVOKE** statement

- It is a tool to manipulate data & perform queries against the database
- It enables you to conduct a conversation with the database
- Two versions of SQL\*PLUS
  - Graphical version
    - Start->All Programs->Oracle Database 10g Express Edition->Go to Database home page(SQL->SQL Commands)
  - Command-Line version
    - Start->All Programs->Oracle Database 10g Express Edition->Run SQL Command Line(Connect)

- A table is an object that can store data in a database
- When you create a table, you must specify the table name, name of each column, data type of each column, and size of each column
- The table and column names can be up to 30 characters long
- Table or column name must begin with a letter
- The names are not case sensitive
- Spaces and hyphens are not allowed in a table or a column name; but \$, \_ and # are allowed

Data type specifies the type of data that will be stored in the column. Data types also help to optimize storage space

- **CHAR(n)**
  - Stores fixed-length alphanumeric data in a column
  - Default and minimum size is one character
  - Maximum allowable size is 2000 characters (previously 255)
  - If a string of a smaller length is stored, it is padded with spaces at the end
- **VARCHAR(n)/VARCHAR2(n)**
  - Stores variable-length alphanumeric data in a column
  - Default and minimum size is one character
  - Maximum allowable size is 4000 characters (previously 2000)
  - If the data are smaller than the specified size, only the data value is stored; no padding is done

[RDBMS](#)[Client/Server Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

- **DATE**

- Stores date and time values
- The range of allowable dates is between January 1, 4712B.C. and December 31, 9999A.D.
- The default date format is DD-MON-YY. The DD-MON-YYYY format also works

- **NUMBER(precision, scale)**

- Stores floating point numbers as well as integer numbers. Precision is the total number of significant digits in the number; scale is the total number of digits to the right of the decimal point(if used). The precision can range from 1 to 38
- If neither precision nor scale is specified, any number may be stored up to a precision of 38 digits

- **INTEGER**

- Stores integer number

- **NUMERIC(p,d)**
  - Stores fixed-point number with user specified precision
  - Similar to NUMBER data type
- **LONG**
  - Stores variable length character strings containing up to 2GB
  - Similar to VARCHAR
  - There can be one LONG data type per table
- **RAW**
  - Stores binary data such as digitized picture or image
  - Maximum allowable size is 2000 Bytes (previously 255 Bytes)
- **LONG RAW**
  - It is the higher range of RAW
  - There can be one LONG data type per table
  - Maximum allowable size is 2GB

- **LOB(Large Object)**
  - Stores large volume of data
  - **BLOB**
    - Used for binary data such as graphics, video clips and audio files up to 4GB
  - **CLOB**
    - Used for character data up to 4GB
  - **BFILE**
    - Stores references to a binary file that is external to the database and is maintained by the operating system's file system

## CREATE TABLE statement

CREATE statement is used for table creation. The syntax is:

```
CREATE TABLE table_name( column datatype,  
column datatype, ... column datatype);
```

For example, create a table for STUDENT ( Roll, Name, Gender, Age, CGPA)

Solution: *CREATE TABLE STUDENT(Roll NUMBER(6), Name VARCHAR2(20), Gender CHAR(1), Age NUMBER(3), CGPA NUMBER(4,2));*



## DESCRIBE statement

DESCRIBE statement is used for viewing table structure. The syntax is:

**DESCRIBE table\_name;** or  
**DESC table\_name;**

For example: *DESCRIBE STUDENT;*

Solution:

Name	Null?	Type
Roll		NUMBER(6)
Name		VARCHAR2(20)
Gender		CHAR(1)
Age		NUMBER(3)
CGPA		NUMBER(4,2)

[RDBMS](#)[Client/Server Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

## Record Insertion

### INSERT statement

INSERT statement is used to insert a new row/record into a table. The syntax is:

**INSERT INTO table\_name (column1, column2,..) VALUES (value1, value2,..);**

- Column names are optional
- Numeric data is not enclosed within quotes; while character and date values are enclosed within single quotes

### Entering NULL values:

- *Implicit method*: Here, column name is omitted from the column list in the INSERT statement
- *Explicit method*: Here, the null value is used as a value for a numeric column, and an empty string (") is used for date or character columns

### Substitution variables

- Substitution variables enable you to create an interactive SQL script, which prompts you to enter a value for the substitution variable
- In command line version, **&** character is used before the substitution variable in the query; whereas **:** character is used in graphical version
- Substitution variables for character and date columns are enclosed within a pair of single quotation marks
- For more records, press **/**
- If an INSERT statement contains a value containing **&** character, it is treated as a substitution variable. In such cases, **SET DEFINE OFF;** and **SET DEFINE ON;** commands are used

[RDBMS](#)[Client/Server Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

## Record Insertion...

- *INSERT INTO STUDENT(Roll, Name, Gender, Age, CGPA) VALUES (705129, 'Uday', 'M', 19, 9.2);*
- *INSERT INTO STUDENT VALUES (705129, 'Uday', 'M', 19, 9.2);*
- *INSERT INTO STUDENT(Roll, Name, CGPA) VALUES (705129, 'Uday', 9.2);*
- *INSERT INTO STUDENT VALUES (&Roll, '&Name', '&Gender', &Age, &CGPA);*
- *INSERT INTO STUDENT (Roll, Name, Gender, Age) VALUES(&Roll, '&Name', '&Gender', &Age);*

### Customized Prompts

ACCEPT command is used for customized prompts. The syntax is:

**ACCEPT variablename PROMPT 'prompt message'**

- ACCEPT Roll PROMPT 'Please enter the Roll of Student:'
- ACCEPT Name PROMPT 'Please enter the Name of Student:'
- ACCEPT Gender PROMPT 'Please enter the Gender of Student:'
- ACCEPT Age PROMPT 'Please enter the Age of Student:'
- ACCEPT CGPA PROMPT 'Please enter the CGPA of Student:'

```
INSERT INTO STUDENT VALUES (&Roll, '&Name',  
'&Gender', &Age, &CGPA);
```

Once a variable is defined with substitution variable or ACCEPT, its value is known throughout that session

[RDBMS](#)[Client/Server  
Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table  
Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

## Time insertion

Time can be inserted by using TO\_DATE()

```
INSERT INTO STUDENT(dob) VALUES (TO_DATE(  
'12-JAN-1990 10:34:45 P.M.', 'DD-MON-YYYY HH:MI:SS  
P.M.'));
```

- If only the date value is entered in a date-type column, the time value is set to the midnight (12:00A.M.)
- If only the time value is entered in a date-type column, the date is set to first of the current month

[RDBMS](#)[Client/Server  
Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table  
Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

## SELECT statement

SELECT statement is used to retrieve data from the underlying table. The syntax is:

**SELECT column1,column2 FROM table\_name;**

If the user wants to see all the columns in a table, \* can be used in place of columns

*SELECT \* FROM STUDENT;*

Roll	Name	Gender	Age	CGPA
705129	Uday	M	19	9.2
705170	Ram	M	20	
705171	Kim	F	19	8.6
705172	Raji		20	7.5

**NULL value** means the value is unknown or doesn't exist

[RDBMS](#)[Client/Server Database](#)[SQL](#)[SQL Statements](#)[SQL\\*PLUS](#)[Naming Conventions](#)[Data Types](#)[Table Creation](#)[Viewing Table Structure](#)[Record Insertion](#)[Inserting Time](#)[Querying Data](#)

*SELECT Roll, Name, CGPA FROM STUDENT;*

Roll	Name	CGPA
705129	Uday	9.2
705170	Ram	
705171	Kim	8.6
705172	Raji	7.5