Unit IV
SQL

Structured Query Language

- Contents
  - SQL Data Definition and Data Types
  - Specifying basic constraints in SQL
  - Schema change statements in SQL
  - Basic queries in SQL
  - More complex SQL Queries
  - Insert, Delete and Update statements in SQL
  - Creating Views Triggers and Stored procedures

- **What is SQL?**
  - SQL stands for Structured Query Language
  - SQL lets you access and manipulate databases

- **What Can SQL do?**
  - SQL can execute queries against a database
  - SQL can retrieve data from a database
  - SQL can insert records in a database
  - SQL can update records in a database
  - SQL can delete records from a database
  - SQL can create new databases
  - SQL can create new tables in a database
  - SQL can create stored procedures in a database
  - SQL can create views in a database
  - SQL can set permissions on tables, procedures, and views

SQL

- Data Definition Language (DDL)
  - Create/alter/delete tables and their attributes
  - Following lectures...
- Data Manipulation Language (DML)
  - Query one or more tables – discussed next !
  - Insert/delete/modify tuples in tables

- **Database Tables**
  - A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

Table name

Attribute names

Tables in SQL

Product

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

Tuples or rows

Tables Explained

- A tuple = a record
  - Restriction: all attributes are of atomic type

- A table = a set of tuples
  - Like a list…
  - …but it is unorderd:
    no **first()**, no **next()**, no **last()**.

Tables Explained

- The *schema* of a table is the table name and its attributes:

Product(PName, Price, Category, Manfacturer)

- A *key* is an attribute whose values are unique; we underline a key

Product(<u>PName</u>, Price, Category, Manfacturer)

- Columns indicate: the Attribute value
- Rows indicate: tuple values
- Each attributes has datatypes.

Data Types in SQL

- Atomic types:
  - Characters: CHAR(20), VARCHAR(50)
  - Numbers: INT, BIGINT, SMALLINT, FLOAT
  - Others: MONEY, DATETIME, …

- Every attribute must have an atomic type
  - Hence tables are flat
  - Why ?

Various Keys:

- **Primary key:**
  - A key which uniquely identifies a tuple in a table is known as Primary key.
- **Composite key:**
  - More than one key is used to identify a unique tuple in a table is known as Composite key
- **Foreign key:**
  - An attribute of one table than refers the primary key of another table is known as foreign key

Data Definition Language

- Here we will study the various operations like
  - CREATE – is used to create table
  - ALTER – is used to modify table
  - DROP – is used to delete table

How to create tables now !!!

- Tables in SQL can be created with the command:
  - Create:
  - **Syntax:**

```
CREATE TABLE STUDENT
(Attribute_Name1  datatype,
Attribute_Name2 datatype,
………
Attribute_Namen datatype
);
```

- Example: WE need to maintain STAFF data of 4th SEM CSE like…

FID	FNAME	LNAME

- First create table with a specific name;
- List the attributes needed along with the datatype;

- Create table staff

```
(
    fid int,
     fname varchar(20),
    Lname varchar(20)
);
```

Things to remember!!!

- Later on you remembered to add one more column value.
- Remember to remove one column value as its extra.
- Gave wrong datatype to the attribute(column).

Alter table Command

- ALTER TABLE statement is used to add, delete, or modify columns in an existing table.
- **To add new column:**
  - Syntax:

```
ALTER TABLE table_name
ADD column_name datatype;
```

- **To drop column:**
  - Syntax:

```
ALTER TABLE table_name
DROP COLUMN column_name ;
```

- **To edit data type:**
- Syntax:

```
ALTER TABLE table_name
MODIFY column_name data_type ;
```

Alter table Command

- ALTER TABLE statement is used to add, delete, or modify columns in an existing table.
- **To add new column:**
  - Syntax:

```
ALTER TABLE table_name
ADD column_name datatype;
```

```
Create table staff
(
      fid int,
       fname varchar(20),
      Lname varchar(20),
);
```

```
ALTER TABLE STAFF
ADD DIV Varchar(10);
```

- After adding column value this is how it looks….

FID	FNAME	LNAME	DIV

Lets add one more column

- Subject they teach is to be added..

```
ALTER TABLE STAFF
ADD SUBJECT Varchar(10);
```

FID	FNAME	LNAME	DIV	SUBJEC T

Lets add one more column

- Hobby of staff

```
ALTER TABLE STAFF
ADD HOBBY Varchar(10);
```

FID	FNAME	LNAME	DIV	SUBJECT	HOBBY

- But hobby is really to do something with the data required??

- NO!!!!!

Better delete the column hobby

- **To delete column:**

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

Better delete the column hobby

- **To delete column:**

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

FID	FNAME	LNAME	DIV	SUBJECT	HOBBY

```
ALTER TABLE STAFF
DROP COLUMN HOBBY;
```

Column deleted !!!!!

FID	FNAME	LNAME	DIV	SUBJEC T

Consider this situation

- Created staff table with this attributes and datatype

```
Create table staff
(
      fid int,
       fname varchar(20),
      Lname int
);
```

- LOOK here LNAME is not a field of datatype int !!!!!

Consider this situation

- Created staff table with this attributes and datatype

```
Create table staff
(
      fid int,
       fname varchar(20),
      Lname int
);
```

- Here you need not require to delete the table for just one mistake…..

Consider this situation

- Created staff table with this attributes and datatype

```
Create table staff
(
	fid int,
	 fname varchar(20),
	Lname int
);
```

- You can change the datatype of the field which u require!!!

Consider this situation

- Created staff table with this attributes and datatype

```
Create table staff
(
      fid int,
       fname varchar(20),
      Lname int
);
```

```
      ALTER TABLE table_name
      MODIFY column_name data_type ;
```

```
      ALTER TABLE staff
      MODIFY LNAME varchar(20) ;
```

If you want to drop entire table

- Command is

  DROP TABLE table_name;

- Example: WE need to maintain STAFF data of 4th SEM CSE like…

FID	FNAME	LNAME	SUBJECT	DIV	PLACE
1	Sanjeev	Sannakki	OS	B	Gokak
2	Vidhya	Kulkarni	MM	B	Belgaum
3	Akshata	Angadi	WEB	B	Hubli
4	Malllikarjun	Math	DAA	B	Belgaum
5	Kuldeep	Sambrekar	DBMS	D	Belgaum
6	Vijay	Rajpurohit	DBMS	A	Bagalkot
7	Padma	Dandannavar	DBMS	C	Belgaum
8	Parimal	Tergundi	DBMS	D	Belgaum

Schema change statements in SQL

- DROP
- ALTER

For example consider this

FID	FNAME	LNAME	SUBJECT	DIV	PLACE
1	Sanjeev	Sannakki	OS	B	Gokak
	Vidhya	Kulkarni	MM	B	Belgaum
3	Akshata	Angadi	WEB	B	Hubli
3	Malllikarjun	Math	DAA	B	Belgaum
	Kuldeep	Sambrekar	DBMS	D	Belgaum
6		Rajpurohit	DBMS	A	Bagalkot
7	Padma	Dandannavar	DBMS	C	Belgaum
8	Parimal	Tergundi	DBMS	D	Belgaum

Specify constraints on table creation

- The available constraints in SQL are:
- **PRIMARY KEY**: A primary key is a field which can uniquely identify each row in a table. And this constraint is used to specify a field in a table as primary key.
- **FOREIGN KEY**: A Foreign key is a field which can uniquely identify each row in a another table. And this constraint is used to specify a field as Foreign key.
- **NOT NULL**: This constraint tells that we cannot store a null value in a column. That is, if a column is specified as NOT NULL then we will not be able to store null in this particular column any more.
- **UNIQUE**: This constraint when specified with a column, tells that all the values in the column must be unique. That is, the values in any row of a column must not be repeated.
- **CHECK**: This constraint helps to validate the values of a column to meet a particular condition. That is, it helps to ensure that the value stored in a column meets a specific condition.
- **DEFAULT**: This constraint specifies a default value for the column when no value is specified by the user.

Example

- CREATE TABLE sample_table

```
(
 column1 data_type(size) constraint_name,
 column2 data_type(size) constraint_name,
 column3 data_type(size) constraint_name, ....
);
```

sample_table: Name of the table to be created.

data_type: Type of data that can be stored in the field. **constraint_name**: Name of the constraint. for example- NOT NULL, UNIQUE, PRIMARY KEY etc.

For example consider this

FID	FNAME	LNAME	SUBJECT	DIV	PLACE
1	Sanjeev	Sannakki	OS	B	Gokak
	Vidhya	Kulkarni	MM	B	Belgaum
3	Akshata	Angadi	WEB	B	Hubli
3	Malllikarjun	Math	DAA	B	Belgaum
	Kuldeep	Sambrekar	DBMS	D	Belgaum
6		Rajpurohit	DBMS	A	Bagalkot
7	Padma	Dandannavar	DBMS	C	Belgaum
8	Parimal	Tergundi	DBMS	D	Belgaum

Observations

- FID column should not be left blank.
- No two people have same fid.

This to be corrected !!!

- First analyze which all fields should not be left empty.
- Secondly analyze which fields should have unique values.
- Then write the create table command !!!!!!!

- 1. NOT NULL keyword should be specified after each attribute which you want not to be left blank.
- 2. PRIMARY KEY should be used to define a key uniquely.

- An attribute which uniquely identify a tuple is known as primary key.

```
Create table staff
(
      fid int,
       fname varchar(20) NOT NULL,
      Lname int,
      Subject varchar(20) NOT NULL,
      Div varchar(5) NOT NULL,
      Place varchar(20),
      Primary key(fid)
);
```

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

Can single attribute be key attribute??????

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

Can single attribute be key attribute?????? NO

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

Can single attribute be key attribute?????? NO

Can combination of attributes be key attribute??????

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

Can single attribute be key attribute?????? NO

Can combination of attributes be key attribute?????? YES

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

Can single attribute be key attribute?????? NO

RollNo and Div

Can combination of attributes be key attribute??????
YES

LIST of all students in CSE dept

Note: Assume each division starts with a 1 as roll number

RollNo	Name	Lname	Div
1	Amit	Patil	A
1	Dilip	Naik	B
3	Anand	Kulkarni	A
2	Amit	Patil	C
6	Samit	Hegde	D
3	Dilip	Patil	B

Which should be key attribute now??????

Can combination of attributes be key attribute?????? YES

Can single attribute be key attribute??????

```
CREATE TABLE STUDENT
(
   Rollno int,
   FNAME   varchar(20) NOT NULL,
    LNAME varchar(20),
    Div varchar(5),
    PRIMARY KEY(Rollno, Div)
);
```

- Assume department table in GIT

DeptName	Location
CSE	FIRST FLOOR
EC	FIRST FLOOR
IS	SECOND FLOOR

- Assume student table listed below

USN	Name	Lname	Dept
1	Amit	Patil	CS
2	Dilip	Naik	EC
3	Anand	Kulkarni	MECH
4	Ganesh	Hegde	IS

- Assume department table in GIT

DeptName	Location
CSE	FIRST FLOOR
EC	FIRST FLOOR
IS	SECOND FLOOR

- Assume student table listed below

USN	Name	Lname	Dept
1	Amit	Patil	CS
1	Dilip	Naik	EC
3	Anand	Kulkarni	MECH
4	Ganesh	Hegde	IS

- Foreign Key: A key attribute of one table referring another table’s attribute is known as foreign key.

```
Create table department
(
      Dname varchar(20),
       Dloc varchar(20) NOT NULL,
      Primary key(Dname),
);
```

```
Create table student
(
      USN varchar(20),
       FNAME varhcar(20) NOT NULL,
      Lname varchar(20),
      Dname varchar(20),
      PRIMARY KEY(USN),
      FOREIGN KEY(Dname) references DEPARTMENT(Dname)
);
```

SQL Constraints

1. **NOT NULL**

If we specify a field in a table to be NOT NULL. Then the field will never accept null value.

That is, you will be not allowed to insert a new row in the table without specifying any value to this field. For example,

the below query creates a table Student with the fields ID and NAME as NOT NULL.

That is, we are bound to specify values for these two fields every time we wish to insert a new row.

```
CREATE TABLE Student
( ID int(6) NOT NULL,
NAME varchar(10) NOT NULL,
ADDRESS varchar(20) );
```

2. UNIQUE

This constraint helps to uniquely identify each row in the table. i.e. for a particular column, all the rows should have unique values. We can have more than one UNIQUE columns in a table.

For example, the below query creates a tale Student where the field ID is specified as UNIQUE. i.e, no two students can have the same ID.

```
CREATE TABLE Student
(
 ID int(6) NOT NULL UNIQUE,
   NAME varchar(10),
 ADDRESS varchar(20)
```

3. PRIMARY KEY

Primary Key is a field which uniquely identifies each row in the table. If a field in a table as primary key, then the field will not be able to contain NULL values as well as all the rows should have unique values for this field. So, in other words we can say that this is combination of NOT NULL and UNIQUE constraints.
A table can have only one field as primary key.

Below query will create a table named Student and specifies the field ID as primary key.

```
CREATE TABLE Student ( ID int(6),
 NAME varchar(10),
ADDRESS varchar(20),
 PRIMARY KEY(ID) );
```

4. CHECK

- Using the CHECK constraint we can specify a condition for a field, which should be satisfied at the time of entering values for this field.
- For example, the below query creates a table Student and specifies the condition for the field AGE as (AGE >= 18).
- That is, the user will not be allowed to enter any record in the table with AGE < 18.

```
  CREATE TABLE Student
 ( ID int(6) NOT NULL,
NAME varchar(10) NOT NULL,
AGE int NOT NULL CHECK (AGE >= 18) );
```

5. DEFAULT

- This constraint is used to provide a default value for the fields. That is, if at the time of entering new records in the table if the user does not specify any value for these fields then the default value will be assigned to them.
- For example, the below query will create a table named Student and specify the default value for the field AGE as 18.

```
CREATE TABLE Student
 ( ID int(6) NOT NULL,
NAME varchar(10) NOT NULL,
AGE int DEFAULT 18 );
```

6. FOREIGN KEY

- Foreign Key is a field in a table which uniquely identifies each row of a another table. That is, this field points to primary key of another table. This usually creates a kind of link between the tables.

- Assume department table in GIT

DeptName	Location
CSE	FIRST FLOOR
EC	FIRST FLOOR
IS	SECOND FLOOR

- Assume student table listed below

USN	Name	Lname	Dept
1	Amit	Patil	CS
2	Dilip	Naik	EC
3	Anand	Kulkarni	MECH
4	Ganesh	Hegde	IS

- Assume department table in GIT

DeptName	Location
CSE	FIRST FLOOR
EC	FIRST FLOOR
IS	SECOND FLOOR

- Assume student table listed below

USN	Name	Lname	Dept
1	Amit	Patil	CS
1	Dilip	Naik	EC
3	Anand	Kulkarni	MECH
4	Ganesh	Hegde	IS

- Foreign Key: A key attribute of one table referring another table’s attribute is known as foreign key.

```
Create table department
(
      Dname varchar(20),
       Dloc varchar(20) NOT NULL,
      Primary key(Dname),
);
```

```
Create table student
(
      USN varchar(20),
       FNAME varhcar(20) NOT NULL,
      Lname varchar(20),
      Dname varchar(20),
      PRIMARY KEY(USN),
      FOREIGN KEY(Dname) references DEPARTMENT(Dname)
);
```