Group by clause

1. To find sum of salary and count number of employees for deptno 20 and 30

```
select deptno,count(*),sum(sal)
from emp
where deptno in (20,30)
group by deptno;
```

2. Find sum for salary of clerks for each department

```
Select deptno,sum(sal)
From emp
Where job='CLERK'
Group by deptno
```

3. Find how many analyst are there in each department

```
Select deptno,count(*)
From emp
Where jpb='ANALYST'
Group by deptno;
```

4. Find sum of netsal for all employees working under each manager
   Netsal=sal+comm

```
Select sum(sal+ifnull(comm,0)),mgr
From emp
Group by mgr;
```

5. How many employees joined in month of may

```
Select count(*)
From emp
Where month(hiredate)=5;
```

6. Display all departments in which more than 2 salesmans are there.

```
Select deptno,count(*)
From emp
Where job='salesman'
Group by deptno
Having count(*)>2
```

To create or modify the table

The commands are DDL commands(Data definition language)

All DDL statements are autocommit

Alter ,create, truncate, drop all these are DDL statements

1. To create a new table

```
Create table student(
  Studid int primary key,
  Sname  varchar(25),
  Address varchar(20));
```

2. Drop the table

This will delete entire table structure and the data

```
Drop table student;
```

3.Truncate table

This will delete only data, but empty table will remain there.

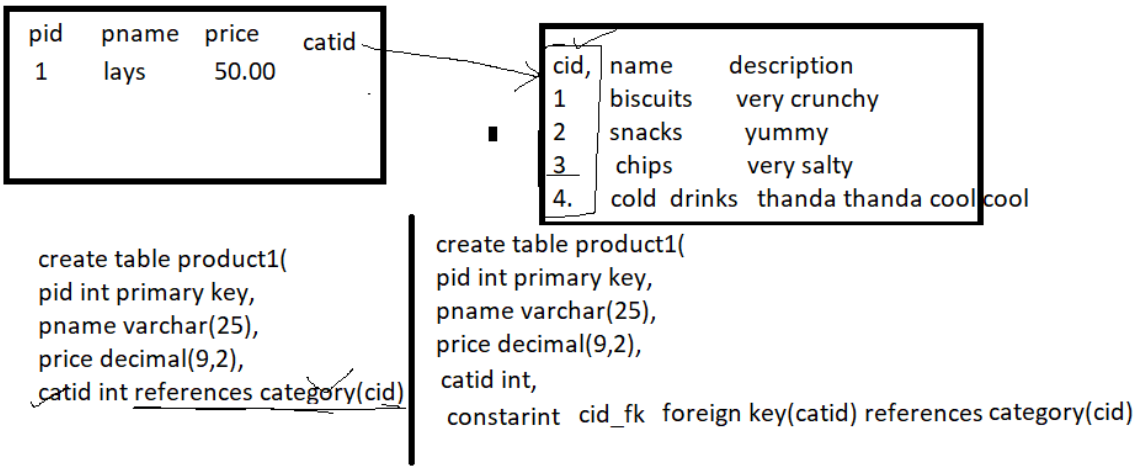
```
Truncate table student;
```

Constraints are available

Primary key	• only one primary key is allowed, it can be simple or composite primary key in a table • Only unique and not null values are allowed • It is table level constraint create table studentIACSD (stud_id int primary key,sname varchar(25),address varchar(20));
Unique	• Any number of unique key constraints are allowed in a table • Only unique values are allowed, • Any number of null values are allowed • It is table level constraint create table studentIACSD(sid int primary key, sname varchar(25), passport_no int unique);
Not null	• Any number of columns can be not null • It is field level constraint • Null values are not allowed create table studentIACSD(sid int primary key, sname varchar(25) not null, passport_no int unique not null);
Default	• Any number of columns you can keep default value

	• Null values are not allowed, because it gets replaced with default value
Check	• Any number of columns can be using check constraint • You can add any valid condition in check constraint • It is table level constraint create table product1 -> (pid int primary key -> pname varchar(20) not null, -> price decimal(9,2) default 100, -> qty int check(qty>0));
Foreign key On delete <action> On update cascade	• Any number of foreign keys can be there in a table • It is table level constraint • It references the value of parent table for correctness of data

- Primary key, foreign key, unique and check constraints are table level constraints
- Not null, default these are field level constraints.



Statement --- create table <table name>(list of columns in the table)

Data types in mysql

Data types in mysql

- Numeric
- Date and Time

- String Types.

Let us now discuss them in detail.

Numeric Data Types

MySQL uses all the standard ANSI SQL numeric data types, so if you're coming to MySQL from a different database system, these definitions will look familiar to you. The following list shows the common numeric data types and their descriptions −

- **INT** − A normal-sized integer that can be signed or unsigned. If signed, the allowable range is from -2147483648 to 2147483647. If unsigned, the allowable range is from 0 to 4294967295. You can specify a width of up to 11 digits.
- **TINYINT** − A very small integer that can be signed or unsigned. If signed, the allowable range is from -128 to 127. If unsigned, the allowable range is from 0 to 255. You can specify a width of up to 4 digits.
- **SMALLINT** − A small integer that can be signed or unsigned. If signed, the allowable range is from -32768 to 32767. If unsigned, the allowable range is from 0 to 65535. You can specify a width of up to 5 digits.
- **MEDIUMINT** − A medium-sized integer that can be signed or unsigned. If signed, the allowable range is from -8388608 to 8388607. If unsigned, the allowable range is from 0 to 16777215. You can specify a width of up to 9 digits.
- **BIGINT** − A large integer that can be signed or unsigned. If signed, the allowable range is from -9223372036854775808 to 9223372036854775807. If unsigned, the allowable range is from 0 to 18446744073709551615. You can specify a width of up to 20 digits.
- **FLOAT(M,D)** − A floating-point number that cannot be unsigned. You can define the display length (M) and the number of decimals (D). This is not required and will default to 10,2, where 2 is the number of decimals and 10 is the total number of digits (including decimals). Decimal precision can go to 24 places for a FLOAT.
- **DOUBLE(M,D)** − A double precision floating-point number that cannot be unsigned. You can define the display length (M) and the number of decimals (D). This is not required and will default to 16,4, where 4 is the number of decimals. Decimal precision can go to 53 places for a DOUBLE. REAL is a synonym for DOUBLE.
- **DECIMAL(M,D)** − An unpacked floating-point number that cannot be unsigned. In the unpacked decimals, each decimal corresponds to one byte. Defining the display length (M)

Date and Time Types

The MySQL date and time datatypes are as follows −

- **DATE** − A date in YYYY-MM-DD format, between 1000-01-01 and 9999-12-31. For example, December 30th, 1973 would be stored as 1973-12-30.
- **DATETIME** − A date and time combination in YYYY-MM-DD HH:MM:SS format, between 1000-01-01 00:00:00 and 9999-12-31 23:59:59. For example, 3:30 in the afternoon on December 30th, 1973 would be stored as 1973-12-30 15:30:00.

- **TIMESTAMP** − A timestamp between midnight, January 1st, 1970 and sometime in 2037. This looks like the previous DATETIME format, only without the hyphens between numbers; 3:30 in the afternoon on December 30th, 1973 would be stored as 19731230153000 (YYYYMMDDHHMMSS).
- **TIME** − Stores the time in a HH:MM:SS format.
- **YEAR(M)** − Stores a year in a 2-digit or a 4-digit format. If the length is specified as 2 (for example YEAR(2)), YEAR can be between 1970 to 2069 (70 to 69). If the length is specified as 4, then YEAR can be 1901 to 2155. The default length is 4.

String Types

Although the numeric and date types are fun, most data you'll store will be in a string format. This list describes the common string datatypes in MySQL.

- **CHAR(M)** − A fixed-length string between 1 and 255 characters in length (for example CHAR(5)), right-padded with spaces to the specified length when stored. Defining a length is not required, but the default is 1.
- **VARCHAR(M)** − A variable-length string between 1 and 255 characters in length. For example, VARCHAR(25). You must define a length when creating a VARCHAR field.

- **BLOB or TEXT** − A field with a maximum length of 65535 characters. BLOBs are "Binary Large Objects" and are used to store large amounts of binary data, such as images or other types of files. Fields defined as TEXT also hold large amounts of data. The difference between the two is that the sorts and comparisons on the stored data are **case sensitive** on BLOBs and are **not case sensitive** in TEXT fields. You do not specify a length with BLOB or TEXT.
- **TINYBLOB or TINYTEXT** − A BLOB or TEXT column with a maximum length of 255 characters. You do not specify a length with TINYBLOB or TINYTEXT.
- **MEDIUMBLOB or MEDIUMTEXT** − A BLOB or TEXT column with a maximum length of 16777215 characters. You do not specify a length with MEDIUMBLOB or MEDIUMTEXT.
- **LONGBLOB or LONGTEXT** − A BLOB or TEXT column with a maximum length of 4294967295 characters. You do not specify a length with LONGBLOB or LONGTEXT.
- **ENUM** − An enumeration, which is a fancy term for list. When defining an ENUM, you are creating a list of items from which the value must be selected (or it can be NULL). For example, if you wanted your field to contain "A" or "B" or "C", you would define your ENUM as ENUM ('A', 'B', 'C') and only those values (or NULL) could ever populate that field.

Vehicle(vid, name,model,chassinumber,cid)

Customer(cid,cname,address)

```
create table customer(
   -> cid int primary key,
   -> cname varchar(20),
   -> address varchar(20));
```

```
Query OK, 0 rows affected (0.04 sec)

mysql> create table vehicle(
    -> vid int primary key,
    -> vname varchar(20),
    -> model varchar(20),
    -> chassienumber int,
    -> custid int,
    -> constraint cid_fk1 foreign key(custid) references customer(cid));
Query OK, 0 rows affected (0.04 sec)

Use on delete cascade
create table vehicle(
vid int primary key,
    vname varchar(20),
   model varchar(20),
    chassienumber int,
    custid int,
   constraint cid_fk1 foreign key(custid) references customer(cid)
   on delete cascade
on update cascade
);


Use on delete set null
create table vehicle(
vid int primary key,
    vname varchar(20),
   model varchar(20),
    chassienumber int,
    custid int,
   constraint cid_fk1 foreign key(custid) references customer(cid)
   on delete no action
on update cascade
);

Create table course


Create table room

Create table course_room
Create table course_room(
cid int,
rid int,
timing datetime,
constraint cr_pk primary key(cid,rid,timing))
```

)

DML operations

1. Insert values for all columns into emp table

Insert into emp values(12,'asd',4567,10)

2. Insert few columns into emp table

Insert into emp(ename,empno) values('asd',12)

3. To update job to CLERK , sal to 12345, comm to 345 for all employees With empno>7902

Update

Update emp
Set job='CLERK',sal=12345,comm=345
Where empno>7902;

4. To delete all employees from deptno 10

Delete
From emp
Where dept=10;