8 days:

**Day 7**

**Database:**

**Join:** Join is use to retrieve more than one column from more than one table with or without conditions.

Cross join or Cartesian product:

Select fewColumnFromFirstTable, fewColumnsFrom secondTable from firstTableName,secondTableName

Result is

M\*N = MN

select first\_name,job\_title from employee, jobs;

cross join using table and column name

select employee.first\_name,employee.job\_id,jobs.job\_title from employee, jobs;

**table alias**

select tableAlias.columName,tableAlias.column from firstName tableAlias,secondTable tableAlias;

select e.first\_name,e.job\_id,j.job\_title from employee e, jobs j;

**Inner join or Equi-Join**

Select fewColumnFromFirstName, fewColumnFromSecondTable firstName inner join secondTableName on firstTableCommonColumn=secondTableCommonColumn

**select e.first\_name,e.salary,d.department\_name from employee e inner join department d on e.department\_id=d.department\_id;**

**select e.first\_name,j.job\_title from employee e inner join jobs j on e.job\_id=j.job\_id;**

left outer join : common + first table remaining records

**select e.first\_name,d.department\_name from employee e left outer join department d on e.department\_id=d.department\_id;**

right outer join : common + second table remaining records

**select e.first\_name,d.department\_name from employee e right outer join department d on e.department\_id=d.department\_id;**

**Employee**

**10 records**

**6 purchase the different insurance.**

**Health insurance**

**5**

**3 company insurance purchased by different employee**

**Order by Clause**

**Which help to display the data in ascending or descending order.**

**select employee\_id, first\_name,salary from employee order by first\_name asc;**

**select employee\_id, first\_name,salary from employee order by first\_name desc**

**select employee\_id, first\_name,salary from employee order by salary asc**

**select employee\_id, first\_name,salary from employee order by salary desc**

**select \* from employee order by hire\_date asc**

**select \* from employee order by hire\_date desc**

**DDL**

**Data Definition language**

**Create :**

**Product -🡪Table Name**

**Pid 🡪 int (primary key)**

**PName -🡪 varchar(50) not null**

**Price 🡪 float (check constraint price must be +ve)**

**Qty -🡪 int default 1**

**create table tableName(columnName dataType1,columName dataType2…ColumName dataTypen);**

**while creating table constraints is optional. Constraint mainly help us to make restriction to insert invalidate data.**

**Primary key: if column is pk that column doesn’t allow duplicate as well as null value. In single table we can create only one column as PK.**

**By default every column can allow null or empty value. If we write not null. We can’t insert null or empty value.**

**Check to apply condition while inserting the records.**

**Default : if we not insert any value by default value consider.**

**Create table product**

**(**

**Pid int primary key,**

**Pname varchar(30) not null,**

**Price float check(price>0),**

**Qty int default 0**

**);**

**DML**

**Insert query**

**Syntax**

**insert into tableName values(v1,v1,v3); : all column name and order must be match.**

**Insert into tableName(c1,c2,c3) values(v1,v2,v3);**

**Update Query**

**Syntax**

**update tableName set ColumnName = value;**

**all record values get updated**

**update product set qty=5;**

**update with where clause**

**update tableName set columnName = value where clause**

**update product set qty=50 where pid=102;**

**update product set pname="Sony TV" where pname='TV';**

**delete query**

**delete from tableName; this query delete all records present in table.**

**Delete with where clause**

**Delete from product where pid=104;**

**Delete from product where pname=”Mobile”;**

**DDL :**

**Create query**

**Drop query : this query use to drop the table. If table contains one or many records it will remove table including all records.**

**Auto\_increment: this keyword help in mysql database to increment the number one by one start with 1.**

**create table task(tid int primary key auto\_increment, task varchar(250));**

**mysql> insert into task(task) values('I am teaching');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> insert into task(task) values('I am typing');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> insert into task(task) values('I am teachning mysql queries');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> select \* from task;**

**mysql> drop table task;**

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

**Delete from tableName; this query remove all records from a table. But still table structure present in database.**

**Drop table tableName; this query remove all records as well as table structure.**

**Alter query**

**We can modify table structure**

1. **Add new column to existing table.**

**alter table product add manifacture\_county varchar(3);**

1. **Modify existing column data types**

**alter table product modify manifacture\_county varchar(10);**

1. **Drop the column**

**Alter table tablename drop column columnName;**

**alter table product drop column qty;**

1. **Rename column name**

**alter table product rename column manifacture\_county to manifacture\_country;**

**truncate : This query is use to delete all records from a table.**

**truncate table tableName;**

**truncate table task; this query remove all records from a table.**

**Truncate Vs delete Vs drop**

1. **Truncate and drop is a part of DDL and delete is a part of DML.**
2. **Truncate and delete remove all records but maintain the table structure. Drop remove all records including table structure.**
3. **With truncate we can’t use where clause but with delete we can use where clause.**
4. **If we use truncate those records deleted permanently but if we delete record using delete query we can rollback those records using TCL query.**