

Database Management System

Unit 5: Database Language - Structured Query Language

Database Language – Structured Query Language

5.1 SQL Fundamentals

5.2 **Data Types of Attributes**

5.3 **DDL Statements in SQL**

5.4 **Constraints in SQL**

5.5 **DML Statements in SQL**

5.6 Data Query Language

5.7 Data Control Language

CE: 5.1

SQL Fundamentals

CE 5.1: SQL Fundamentals

SQL – Structured Query Language

- SQL is a language that enables you to create and operate on relational database, which are sets of related information stored in tables.
- SQL statements are like plain English statements, but with a specific syntax.
- It is simple and powerful language used to create, access and manipulate data for structuring database.

CE 5.1: SQL Fundamentals

➤ SQL Statement Categories:

1. Data Definition Language
2. Data Manipulation Language
3. Data Query Language
4. Data Control Language

CE 5.1: SQL Fundamentals

1) Data Definition Language:

- A database schema is specified by a set of definitions which are expressed by a special language called a ***data definition language (DDL)***.
- The compilation result is a set tables which are stored in a special file called *data dictionary*.
- DDL provides commands like.....

Create, Alter, Drop, Truncate and Comment.

CE 5.1: SQL Fundamentals

2) Data Manipulation Language:

- After the database scheme has been specified and the database has been created, the data can be manipulated using a set of procedures which are expressed by a special language called a *data manipulation language (DML)*.
- A DML is a language that enables users to access or to manipulate data within database tables.
- DML provides commands like.....

Insert, Update and Delete.

CE 5.1: SQL Fundamentals

3) Data Query Language:

- The commands that are used to retrieve data from the database table(s) and also to perform operations on those data are called as *Data query language (DQL)*.
- **DQL provides commands like.....**

Select.

CE 5.1: SQL Fundamentals

4) Data Control Language:

- *Data control language (DCL)* is a SQL syntax, which is used to control access to the database and table data.

OR

- *DCL* is mainly used to deal with the rights, permissions and other controls of the database system and also transaction within the database.
- DCL statements are grouped with DML statements.
- **DCL provides commands like.....**

Commit, Savepoint, Rollback, Set Transaction,

Grant/Revoke.

Home Work

1. What are the different types of statements supported by SQL? [1 Mark]
2. Explain DML and DDL with proper examples. [5 Marks]
3. Discuss SQL statements in detail. [5 Marks]

CE: 5.2

Data Types of Attributes

CE 5.2: Data Types of Attributes

Oracle Built-in Datatypes

Category	Data Types
Character	char, nchar, varchar, varchar2, nvarchar2
Number	number, float, binary_float, binary_double
Long and raw	long, long raw, raw
Date and time	date, timestamp, timestamp with time zone, timestamp with local time zone, interval year to month, interval day to second
Large object	clob, nclob, blob, bfile
Row ID	rowid, urowid

CE 5.2: Data Types of Attributes

MySQL Built-in Datatypes

Types	Data Types
Text	char, varchar, blob, text, binary and varbinary
Number	int, float, decimal, double
Date	date, datetime, timestamp, year, time
Misc	enum, set

1. Text Types

1. CHAR(<size>):

- It is used to store character strings values of **fixed-length**.
- It can hold maximum 255 characters.
- The data stored in CHAR columns is **right-padded** with spaces to whatever length specified.
- The default size is 1.

1. Text Types (Conti...)

2. VARCHAR(<size>):

- It is used to store alphanumeric strings values of **variable-length**.
- It can hold 1 to 255 characters.
- It usually a wiser choice, than CHAR, due to it's variable length format characteristic.
- But, the CHAR is 50% faster, than VARCHAR.
- There is **no default size**.

1. Text Types (Conti...)

Difference between CHAR and VARCHAR2:

- In CHAR datatype, trailing spaces are ignored and VARCHAR datatype, trailing spaces are not ignored, and so they sort higher than no trailing spaces.

- Example:

CHAR datatype: 'Yo' = 'Yo '

VARCHAR2 datatype: 'Yo' < 'Yo '

2. Number Types

1. `int(<size>)`

- It is used to represent a number without a decimal point.
- It can be segmented into five types such as...
 - `tinyint`,
 - `smallint`,
 - `mediumint`
 - `int` and
 - `bigint`.

2. Number Types

1. `int(<size>)` (Conti...)

- If unsigned, then the allowable range is for ...
 - `tinyint` – 0 to 255, width is up to 4 digits.
 - `smallint` – 0 to 65535, width is up to 5 digits.
 - `mediumint` – 0 to 16777215, width is up to 9 digits.
 - `int` – 0 to 4294967295, width is up to 11 digits. [default is 11]
 - `bigint` – 0 to 18446744073709551615, width is up to 20 digits.

2. Number Types (Conti...)

2. float(<precision>,<scale>)

- It is used to accept small decimal numbers.
- The values inserted into a table column of the FLOAT type are rounded.
- It cannot be unsigned. You can define the display length (p) and the number of decimals (s).
- The default value is 10,2; where 10 is the total number of digits (including decimals) and 2 is the number of decimals.

2. Number Types (Conti...)

3. decimal(<p>,<s>)

- It is used to store business data values where it is important to preserve exact precision (p).
- The values inserted into a table column of the DECIMAL type are not rounded.
- It cannot be unsigned. You can define the display length (p) and the number of decimals (s).

2. Number Types (Conti...)

4. `double(<p>,<s>)`

- It is used to store large number with a float decimal point.
- The values inserted into a table column of the DOUBLE type are not rounded.
- It cannot be unsigned. You can define the display length (p) and the number of decimals (s).
- The default value is 16,4; where 16 is the total number of digits (including decimals) and 4 is the number of decimals.

3. Date Types

1. date

- It is used to hold dates.
- The standard format to store date is YYYY-MM-DD.
- It ranges from '1000-01-01' to '9999-12-31'.
- To enter dates other than the standard format is, to use appropriate date functions.

3. Date Types (Conti...)

2. datetime

- It is used to hold date and time values.
- The standard format to store date and time is YYYY-MM-DD HH:MM:SS.
- It ranges from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.

3. Date Types (Conti...)

3. timestamp

- It is also used to hold date and time values.
- Values are converted from the current time zone to UTC (**Universal Time Coordinated**) while storing and converted back from UTC to the current time zone when retrieved.
- The standard format to store date and time is YYYY-MM-DD HH:MM:SS.
- It ranges from '1970-01-01 00:00:01' UTC to '2038-01-19 03:14:07' UTC.

3. Date Types (Conti...)

3. Timestamp (Conti...)

- Note:

MySQL does not accept timestamp values that includes a zero in the day or month column or values that are not a valid date.

3. Date Types (Conti...)

4. year

- It is a one-byte data type used for representing years.
- MySQL retrieves and displays YEAR values in YYYY format.
- Its range is from 1901 to 2155.
- It can be declared as YEAR(2) or YEAR(4) to specify a display width of 2 or 4 characters.
- The default width is 4 characters.

3. Date Types (Conti...)

4. year (Conti...)

- YEAR values can be specified in a variety of formats:

String length	Range
4-digit string	'1901' to '2155'.
4-digit number	1901 to 2155.
1- or 2-digit string	'0' to '99'. Values in the ranges '0' to '69' and '70' to '99' are converted to YEAR values in the ranges 2000 to 2069 and 1970 to 1999.
1- or 2-digit number	1 to 99. Values in the ranges 1 to 69 and 70 to 99 are converted to YEAR values in the ranges 2001 to 2069 and 1970 to 1999.

3. Date Types (Conti...)

5. time

- It is used to hold time values in a HH:MM:SS format.
- Data can be assigned to the TIME column by using either a string or a number.

Home Work

1. List at least six datatypes supported by MySQL.

[2 Marks]

2. What is the difference between CHAR and VARCHAR?

[2 Marks]

CE: 5.3

DDL Statements in SQL

CE 5.3: DDL Statements in SQL

Rules for creating tables:

- A table name can have maximum of 30 characters.
- Alphabets from A-Z, a-z and numbers from 0-9 are allowed.
- A table name should begin with an alphabet.
- Special character like **(\$), (_, and (#)**, can be allowed in names and also recommended.
- SQL reserved words not allowed. For example: create, select and so on.

CE 5.3.1: Creating Tables

- Syntax:

```
create table <TableName> (  
    <ColumnName1> <DataType> (<Size>),  
    <ColumnName2> <DataType> (<Size>), .....);
```


CE 5.3.1: Creating Tables (Conti...)

- For example:

```
create table tblStudent(  
  enro int,  
  fname varchar(20),  
  lname varchar(20),  
  city varchar(15),  
  gender char(1),  
  dob date,  
  contactno bigint(10));
```

CE 5.3.2: Modifying Tables

❑ Adding Columns:

1. To add new columns:

Syntax:

```
alter table <TableName> add (<NewColumnName> <Datatype> (<Size>),  
                             <NewColumnName> <Datatype> (<Size>), .....);
```

2. To add new columns at a specific location:

Syntax:

```
alter table <TableName> add <NewColumnName> <Datatype> (<Size>)  
after <OldColumnName>;
```

3. To add new columns as the first column:

Syntax:

```
alter table <TableName> add <NewColumnName> <Datatype> (<Size>)  
first;
```

CE 5.3.2: Modifying Tables (Conti...)

❑ Modifying Columns:

1. To modify existing columns:

Syntax:

```
alter table <TableName>  
modify (<ColumnName> <NewDatatype> (<NewSize>));
```

2. To modify existing column and shift at a specific location:

Syntax:

```
alter table <TableName>  
modify <ColumnName> <NewDatatype> (<NewSize>) after <ColumnName>;
```

3. To modify an existing column and make it the first column:

Syntax:

```
alter table <TableName>  
modify <ColumnName> <NewDatatype> (<NewSize>) first;
```

CE 5.3.2: Modifying Tables (Conti...)

❑ Renaming Columns:

1. To rename existing columns:

Syntax:

Alter table <TableName>

change <OldColumnName> <NewColumnName> <Datatype> (<Size>;

2. To rename an existing column and shift at a specific location:

Syntax:

Alter table <TableName>

change <OldColumnName> <NewColumnName> <Datatype> (<Size>

after <ColumnName>;

3. To rename an existing column and make it the first column:

Syntax:

Alter table <TableName>

change <OldColumnName> <NewColumnName> <Datatype> (<Size>) **first**;

CE 5.3.2: Modifying Tables (Conti...)

❑ Dropping Columns:

1. To drop any particular column:

Syntax:

```
alter table <TableName> drop column <ColumnName>;
```

CE 5.3.2: Modifying Tables (Conti...)

❑ Renaming Tables:

1. To rename a table name:

Syntax:

Alter table <OldTableName> **rename** <NewTableName>;

OR

rename table <OldTableName> **to** <NewTableName>;

CE 5.3.2: Modifying Tables (Conti...)

❑ Dropping Tables:

1. To drop table:

Syntax:

`drop table <TableName>;`

OR

`drop table [if exists] <TableName>;`

CE 5.3.3: Truncating Tables

- The **TRUNCATE TABLE** command is used to clear given table completely, i.e. it removes all the records from the given table.

- Syntax:

truncate table <TableName>;

- For example:

truncate table tblStudent;

CE 5.3.4: Comment

- In MySQL, comment can be given in two ways:
 1. Single line comment, by using -- and #.
 2. Multiple line comment, by using /* */.
- For example:
 - ✓ desc student; -- To display the structure of student table.
 - ✓ desc student; # To display the structure of student table.
 - ✓ create table student(
 enro int,
 fname varchar(20),
 lname varchar(20),
 /* address varchar(50),
 area varchar(15),*/
 contact bigint);

Home Work

1. State the usage of DDL statements. [1 Mark]
2. Which are the commands of DDL statement? [1 Mark]
3. How can we comments in SQL? [1 Mark]
4. Write the rules to create tables in SQL. [2 Marks]
5. How to create table in MySQL? Give example. [2 Marks]
6. Distinguish TRUNCATE and DROP statement. [2 Marks]

CE: 5.4

Constraints in SQL

CE 5.4: Constraints in SQL

- **Constraints** are used to define rules to allow or to restrict what values can be stored in columns.
- The purpose of inducing constraints is to apply the integrity of a database.
- Integrity constraints prevent bad data from being entered into the database.
- They are used to limit the type of data that can be inserted into a table.

CE 5.4: Constraints in SQL (Conti...)

- It can be classified into two types:
 1. column level
 2. table level
- The **column level constraints** can apply only to one column where as **table level constraints** are applied to the entire table.

CE 5.4: Constraints in SQL (Conti...)

Creating Constraints:

- Constraints are created using the **CREATE TABLE** or **ALTER TABLE** statements.
- Constraint can be applied at **column level** if the constraint is defined on a **single column**.
- **Multiple column** constraints must be defined at the **table level**.

CE 5.4: Constraints in SQL (Conti...)

Types of Data Constraints:

1. NOT NULL constraint
2. UNIQUE constraint
3. PRIMARY KEY constraint
4. FOREIGN KEY constraint
5. CHECK constraints
6. DEFAULT constraints

CE 5.4: Constraints in SQL (Conti...)

1. NOT NULL Constraint:

- A NOT NULL constraint is used to define a mandatory column, which ensures that a column cannot have a NULL value.
- **It can be defined at the column level** for the multiple-column; but it cannot be defined at the table level.
- Syntax:

<ColumnName> <Datatype> (<Size>) NOT NULL

CE 5.4: Constraints in SQL (Conti...)

1. NOT NULL Constraint: (Conti...)

- For example:

```
create table tblStudent(  
  enro int,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) );
```

CE 5.4: Constraints in SQL (Conti...)

2. UNIQUE Constraint:

- A UNIQUE constraint is used to ensure that, no two rows should contain duplicate data in the specified column(s).
- **It can be defined at the column level** for a single-column and also **at the table level** for the multiple-column. *Thus, a unique key, should be defined at the table level.*
- Column-level syntax:

<ColumnName> <Datatype> (<Size>) **UNIQUE**

CE 5.4: Constraints in SQL (Conti...)

2. UNIQUE Constraint: (Conti...)

- Table-level syntax:

```
Create table <TableName>(  
<ColumnName1> <DataType> (<Size>),  
<ColumnName2> <DataType> (<Size>), ..... ,  
UNIQUE (<ColumnName1>, <ColumnName2>));
```

- The maximum number of columns specified can be 16.

CE 5.4: Constraints in SQL (Conti...)

2. UNIQUE Constraint: (Conti...)

- For example:

```
create table tblStudent(  
  enro int,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE );
```

```
create table tblStudent(  
  enro int,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10),  
  emailid varchar(30),  
  UNIQUE (contactno, emailid));
```

CE 5.4: Constraints in SQL (Conti...)

2. UNIQUE Constraint: (Conti...)

- For example:

By assigning user defined names to constraints...

```
create table tblStudent(  
  enro int,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10),  
  emailid varchar(30),  
  CONSTRAINT UC_tblStudent UNIQUE (contactno, emailid));
```

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint:

- A PRIMARY KEY constraint is a combination of a NOT NULL and UNIQUE constraint, which is used to uniquely identify a single row in the table.
- It cannot allow duplicate or NULL values.
- A primary key is not compulsory, but it is recommended.
- **Only one primary key is allowed per table.** (i.e. it can be applied on only one column of the table)

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- But, it can be held on more columns in a table, if required.
- A **single column primary key** is called a **simple** primary key.
- A **multicolumn primary key** is called a **composite** primary key.

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- Column-level syntax:

<ColumnName> <Datatype> (<Size>) **primary key**

- Table-level syntax:

primary key (<ColumnName>, <ColumnName>)

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- For example:

```
create table tblStudent(  
  enro int PRIMARY KEY,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE );
```

```
create table tblStudent(  
  enro int PRIMARY KEY,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE,  
  emailid varchar(30),  
  PRIMARY KEY (enro, contactno));
```

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- For example:

By assigning user defined names to constraints...

```
create table tblStudent(  
  enro int PRIMARY KEY,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE,  
  emailid varchar(30),  
  CONSTRAINT PK_tblStudent PRIMARY KEY (enro,  
  contactno));
```

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- The value of primary key column can be also auto generated by the system using “**AUTO_INCREMENT**”.
- Syntax for auto generating value:

```
Create table <TableName>(
<ColumnName1> <Datatype> (<Size>) AUTO_INCREMENT,
<ColumnName2> <DataType> (<Size>), ..... ,
PRIMARY KEY (<ColumnName1>));
```

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- For example:

```
create table tblStudent(  
  enro int AUTO_INCREMENT,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE,  
  PRIMARY KEY (enro));
```


- By default, the starting value for **AUTO_INCREMENT** is 1, and it will increment by 1 for each new record.

CE 5.4: Constraints in SQL (Conti...)

3. PRIMARY KEY Constraint: (Conti...)

- For example:

```
create table tblStudent(  
  enro int PRIMARY KEY AUTO_INCREMENT,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE );
```



**This will also
work.**

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint:

- A FOREIGN KEY represent the relationships between tables.

or

- A FOREIGN KEY is a key used to link two tables together.
- It is a column (or a group of columns) whose values are derived from the primary key or a unique key of some other table.
- The table in which the foreign key is defined is called a **Foreign table** or **Detail table**.

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- The table that defines the primary or unique key and is referenced by the foreign key is called the **Primary table** or **Master table**.
- The master table's, primary key column is referenced by the detail table's foreign key by using “**references**” **clause**, while defining the foreign key column and its attributes in the detail table.

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- Column-level syntax:

<ColumnName> <Datatype> (<Size>) **references**
 <TableName> (<ColumnName>)

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- For example:

```
create table tblDepartment(  
  did int primary key,  
  dname varchar(50));
```

**Primary table
or
Master table.**

```
create table tblStudent(  
  enro int primary key,  
  fname varchar(20) not null,  
  lname varchar(20) not null,  
  contactno bigint(10) unique,  
  departmentid int references tblDepartment(did));
```

**Foreign table
or
Detail table**

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- For example:

```
create table tblDepartment(  
  did int primary key,  
  dname varchar(50));
```

```
create table tblStudent(  
  enro int primary key,  
  fname varchar(20) not null,  
  lname varchar(20) not null,  
  contactno bigint(10) unique,  
  departmentid int,  
  foreign key (departmentid) references tblDepartment(did));
```

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- For example:

By assigning user defined names to constraints...

```
create table tblDepartment(  
  did int primary key,  
  dname varchar(50));
```

```
create table tblStudent(  
  enro int primary key,  
  fname varchar(20) not null,  
  lname varchar(20) not null,  
  contactno bigint(10) unique,  
  departmentid int,  
  constraint FK_tblStudent foreign key (departmentid)  
  references tblDepartment(did));
```

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- For example:

```
create table tblInstitute(  
iid int primary key,  
iname varchar(50));
```

**Primary table
or
Master table.**

```
create table tblDepartment(  
did int primary key,  
dname varchar(50),  
instituteid int references tblInstitute(iid));
```

**Foreign table
or
Detail table**

CE 5.4: Constraints in SQL (Conti...)

4. FOREIGN KEY Constraint: (Conti...)

- The “foreign key” in the detail table can hold duplicate values.
- It is used to prevent actions that would destroy links between tables.

CE 5.4: Constraints in SQL (Conti...)

5. CHECK Constraint:

- A CHECK constraint is used to ensure that all values in a column satisfies a specific condition.
- It can be defined at the column-level or table-level.

CE 5.4: Constraints in SQL (Conti...)

5. CHECK Constraints: (Conti...)

- Column-level syntax:

<ColumnName> <Datatype> (<Size>) CHECK (<Condition>)

- In condition/Logical Expression, the limits values can be inserted into a column of a table.

- For example:

```
create table tblStudent2(  
  enro int PRIMARY KEY,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE,  
  per float(4,2) CHECK (per < 100));
```

CE 5.4: Constraints in SQL (Conti...)

5. CHECK Constraints: (Conti...)

- Table-level syntax:

CHECK (<Condition>)

- For example:

```
create table tblStudent(  
  enro int PRIMARY KEY,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  contactno bigint(10) UNIQUE,  
  per float(4,2),  
  CHECK (per < 100));
```


CE 5.4: Constraints in SQL (Conti...)

5. CHECK Constraints: (Conti...)

- For example:

By assigning user defined names to constraints...

```
create table tblStudent(  
  enro int PRIMARY KEY,  
  fname varchar(20) NOT NULL,  
  lname varchar(20) NOT NULL,  
  city varchar(15) NOT NULL,  
  contactno bigint(10) UNIQUE,  
  per float(4,2),  
  CONSTRAINT CH_tblStudentper CHECK (per < 100));
```

CE 5.4: Constraints in SQL (Conti...)

6. DEFAULT Constraint:

- A DEFAULT constraint is used to set a default value for a column.
- When a user does not enter a value for the column, automatically the defined default value is inserted in the field.
- The datatype of the default value should match the datatype of the column.
- **It can be defined at the column-level.**

CE 5.4: Constraints in SQL (Conti...)

6. DEFAULT Constraint: (Conti...)

- Column-level syntax:

<ColumnName> <Datatype> (<Size>) DEFAULT (<Value>)

- For example:

```
create table tblAccount(  
  account_no char(10) PRIMARY KEY,  
  holder_name varchar(20),  
  balance decimal(8,2) DEFAULT 2000);
```

CE 5.4: Constraints in SQL (Conti...)

❑ Adding Constraints:

- Constraints can be added in existing table(s), if it is left while creating table.
- They are added using the **ALTER TABLE** statement.
- Syntax:

Alter table <TableName> **add**
<Constraint> <ColumnName1, ColumnName2,...>;

CE 5.4: Constraints in SQL (Conti...)

❑ Adding Constraints: (Conti...)

▪ For examples:

alter table tblStudent add unique (contactno, emailid);

alter table tblStudent add primary key (enro, contactno);

**alter table tblStudent add foreign key (departmentid)
references tblDepartment(did);**

alter table tblStudent add check (per < 100);

CE 5.4: Constraints in SQL (Conti...)

❑ Adding Constraints: (Conti...)

▪ For examples:

```
alter table tblStudent add constraint UC_tblStudent  
unique (contactno, emailid);
```

```
alter table tblStudent add constraint PK_tblStudent  
primary key (enro, contactno);
```

```
alter table tblStudent add constraint FK_tblStudent  
foreign key (departmentid) references tblDepartment(did);
```

```
alter table tblStudent add constraint CH_tblStudentper  
check (per < 100);
```

CE 5.4: Constraints in SQL (Conti...)

❑ Adding Constraints: (Conti...)

▪ For examples:

```
alter table tblAccount alter balance set default 2000;
```

```
alter table tblAccount alter column balance set default  
2000;
```

CE 5.4: Constraints in SQL (Conti...)

❑ Dropping Constraints:

- Constraints are dropped using the **ALTER TABLE** statement.
- Any constraint can be dropped by specifying the constraint name.
- Syntax:

Alter table <TableName> **drop**
<Constraint> <ColumnName1, ColumnName2,...>;

CE 5.4: Constraints in SQL (Conti...)

❑ Dropping Constraints: (Conti...)

▪ For examples:

```
alter table tblStudent drop unique UC_tblStudent;
```

```
alter table tblStudent drop primary key;
```

```
alter table tblStudent drop foreign key FK_tblStudent;
```

```
alter table tblStudent drop check CH_tblStudentper;
```

```
alter table tblStudent drop constraint UC_tblStudent;
```

```
alter table tblStudent drop constraint PK_tblStudent;
```

CE 5.4: Constraints in SQL (Conti...)

❑ Dropping Constraints: (Conti...)

▪ For examples:

```
alter table tblStudent drop constraint FK_tblStudent;
```

```
alter table tblStudent drop constraint CH_tblStudentper;
```

```
alter table tblAccount alter balance drop default;
```

```
alter table tblAccount alter column balance drop default;
```

Home Work

1. Define constraint. [1 Mark]
2. What are the various levels of constraints? [1 Mark]
3. What is a Unique key? [1 Mark]
4. What is a primary key? [1 Mark]
5. What is a composite primary key? [1 Mark]
6. State the use of Auto Increment. [1 Mark]
7. What are foreign keys? [1 Mark]
8. What is CHECK Constraint? [1 Mark]
9. Why do we use SQL constraints? Which constraints we can use while creating database in SQL? [2 Marks]
10. Can a table contain multiple primary key's? Justify your answer. [2 Marks]
11. Write the difference between UNIQUE and PRIMARY KEY constraint. [2 Marks]

Home Work

12. Is it possible for a table to have more than one foreign key? [2 Marks]

CE: 5.5

DML Statements in SQL

CE 5.5: DML Statements in SQL

- **Data manipulation language (DML) statements are used for manipulating following operation:**
 1. **Insert** – to add a new record(s) in a table.
 2. **Update** – to modify existing records values stored in a table.
 3. **Delete** – to remove existing records from the table.

CE 5.5.1: Inserting Rows

❑ To insert/add new records in a table:

- Insert statement can be written in two ways:

1) By specifying both the column names and the values:

Syntax:

```
insert into <TableName> (ColumnName1, ColumnName2,  
ColumnName3, ...)  
values (Value1, Value2, Value3, ...);
```

2) By adding values for all the columns of the table:

[You do not need to specify the column names in the SQL query.]

Syntax:

```
insert into <TableName> values (Value1, Value2, Value3, ...);
```

CE 5.5.1: Inserting Rows (Conti...)

- ❑ Inserting data into a table from another table:

Syntax:

```
Insert into <DestinationTableName> (ColumnName1,  
ColumnName2, ..., ColumnNameN)  
select <ColumnName1, ColumnName2, ..., ColumnNameN> from  
<SourceTableName> where <Condition>;
```


CE 5.5.2: Updating Rows

- ❑ To modify existing records in a table:

Syntax:

```
update <TableName> set <ColumnName1> = <Value1>,  
<ColumnName2> = <value2>, ...  
where <Condition>;
```

CE 5.5.3: Deleting Rows

- ❑ To remove/delete existing records from the table:

Syntax:

delete from <TableName> where <Condition>;

- ❑ To delete all records from the table:

Syntax:

delete from <TableName>;

CE: 5.6

Data Query Language

CE 5.6.1: Selecting Rows

- ❑ To display/retrieve specific records from the database table:

Syntax:

```
select * from <TableName> where <Condition>;
```

- ❑ To display/retrieve all records from the table:

Syntax:

```
select * from <TableName>;
```

CE 5.6.2: Using “select” Statement

❑ Column Alias Name:

It is defined next to the column name with a space or by using the keyword “**AS**”.

For example:

```
select fname 'First Name', lname 'Last Name'  
from tblStudent;
```

```
select fname as 'First Name', lname as 'Last Name'  
from tblStudent;
```

CE 5.6.2: Using “select” Statement

❑ Ensuring Uniqueness:

The “**DISTINCT**” keyword ensures that resulting rows are unique.

For example:

```
select distinct (city) from tblStudent;
```

CE 5.6.3: Limiting Rows

- To limit the number of rows, **WHERE clause** is used.
- The WHERE clause is used to filter records.
- The WHERE clause is used to extract only those records that fulfill a specified condition.
- Any logical conditions of WHERE clause use the **comparison operators** (like =, <>, >, <, >=, <=, between, like, in).

For example:

```
select * from tblStudent where city='Surat';
```

CE 5.6.4: Sorting Rows

- The SELECT statement, include the **ORDER BY clause** is used to sort the result-set in ascending or descending order.
- It sorts the records in ascending order by default or by using ASC keyword.
- To sort the records in descending order, use the DESC keyword.

For example:

```
select * from tblStudent order by fname;
```

```
select * from tblStudent order by fname desc;
```


CE 5.6.5: Selecting records from the beginning or ending of a result set

MySQL supports the **LIMIT clause**, to select a limited number of records from the table.

For example:

```
select * from tblStudent limit 5; # retrieve first 5 rows
```

```
select * from tblStudent limit 2, 5; # skip 2 rows and  
retrieve next 5 rows
```

```
select * from tblStudent where city='Surat' limit 5; #  
retrieve first 5 row those city is "Surat"
```

```
select * from tblStudent order by enro limit 5; # retrieve  
first 5 rows
```

```
select * from tblStudent order by enro desc limit 5; #  
retrieve last 5 rows
```

CE 5.6.6: Expressions

- An expression is a combination of **one or more values**, **operators**, and **SQL Functions** that result in a value.
- **Following expressions can appear in queries:**
 - **SELECT clause** of query
 - **WHERE clause**, **ORDER BY clause**, **LIMIT clause**, **HAVING clause** and **Group By clause**
 - **VALUES clause** of INSERT statement
 - **SET clause** of UPDATE statement

Home Work

1. What is the main usage of DML statements? [1 Mark]
2. Which are different Clauses used in SQL? [1 Mark]
3. How can we avoid duplicating records in a query?

or

3. How to select unique records from a table? [1 Mark]
4. How can we sort data in SQL? [1 Mark]
5. How can we retrieve top 5 records from a table? [1 Mark]
6. Can we rename a column in the output of SQL query?
How? [2 Marks]
7. Difference between Rename and Alias. [2 Marks]
8. What is the difference between DELETE and
TRUNCATE statement? [2 Marks]

CE: 5.7

Data Control Language

CE 5.7: Data Control Language

- In DBMS, a **transaction** is a sequence of related instruction that must be treated as one indivisible unit.
- This means, that **all the work in the transaction must be done or all of it must be undone.**
- This concept is know as **atomicity.**
- Hence, a series of one or more SQL statements that are logically related or a series of operations performed on MySQL table data is called as a **transaction.**

CE 5.7: Data Control Language (Conti...)

- A transaction can include one or more DML statements.
- MySQL treats changes to the table data in two steps:
 - 1) The changes requested by the uses are done, and can be closed by using **COMMIT**.
 - 2) The changes which are done by the user can undone or undo, by using **ROLLBACK**.
- Therefore, by using “**commit**” you can save the current transaction and by using “**rollback**” you can undo the changes made during the transaction.

CE 5.7: Data Control Language (Conti...)

- A transaction is a group of events that occur between any of the following events:
 - 1) Connecting to MySQL
 - 2) Disconnecting to MySQL
 - 3) Committing changes to the database table
 - 4) Rollback

CE 5.7: Data Control Language (Conti...)

- The following statements provide control over the use of transactions in **MySQL**:

COMMIT	Used to permanently save any transaction into the database.
START TRANSACTION	Used to start a new transaction.
ROLLBACK	Restores database table data or undoes all the data changes in the current transaction.

CE 5.7: Data Control Language (Conti...)

```
mysql> select * from tblstudent;
```

enro	fname	lname	contactno	emailid
101	Bhavik	Sarang	9865320147	bhavik@gmail.com
102	Krishna	Patel	9753159620	krishna@gmail.com
104	Om	Puri	8563297410	om@gmail.com
105	Rony	Chauhan	6548520371	rony@utu.ac.in
103	Shyam	Mehera	7418529630	shyam@yahoo.com

```
5 rows in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> delete from tblstudent where enro=105;
Query OK, 1 row affected (0.03 sec)
```

CE 5.7: Data Control Language (Conti...)

- Conti...

```
mysql> select * from tblstudent;
```

enro	fname	lname	contactno	emailid
101	Bhavik	Sarang	9865320147	bhavik@gmail.com
102	Krishna	Patel	9753159620	krishna@gmail.com
104	Om	Puri	8563297410	om@gmail.com
103	Shyam	Mehera	7418529630	shyam@yahoo.com

```
4 rows in set (0.00 sec)
```

```
mysql> rollback;
```

```
Query OK, 0 rows affected (0.08 sec)
```

```
mysql> select * from tblstudent;
```

enro	fname	lname	contactno	emailid
101	Bhavik	Sarang	9865320147	bhavik@gmail.com
102	Krishna	Patel	9753159620	krishna@gmail.com
104	Om	Puri	8563297410	om@gmail.com
105	Rony	Chauhan	6548520371	rony@utu.ac.in
103	Shyam	Mehera	7418529630	shyam@yahoo.com

```
5 rows in set (0.00 sec)
```

CE 5.7: Data Control Language (Conti...)

- Therefore, COMMIT whenever you exit out from MySQL or Oracle.
- Abnormal termination, such as closing the window, can rollback the transaction.
- Machine failure or database crash can also rolled back the transaction.
- **Some statements cannot be roll back like DDL statements such as create or drop databases, tables and so on.**

CE 5.7: Data Control Language (Conti...)

- The following statements provide control over the use of transactions in **Oracle**:

COMMIT	Used to permanently save any transaction into the database.
SAVEPOINT	Used to identify a point in a transaction to which you can later roll back.
ROLLBACK	Used to restore database table data or undoes all the data changes in the current transaction.
SET TRANSACTION	Used to set transaction options, e.g. what roll back segment to use.
GRANT/REVOKE	Grant or revoke (take back) user permissions on a database schema.

CE 5.7: Data Control Language (Conti...)

```
SQL> select * from tblstudent;
```

ENRO	FNAME	LNAME	CONTACTNO	EMAILID
1	Bhavik	Sarang	9857463210	bhavik@gmail.com
2	Krishna	Patel	8965320147	krishna@gmail.com
3	Om	Puri	9658741230	om@yahoo.com
4	Raj	Chauhan	6523987410	raj@yahoo.com
5	Vikas	Patil	7896541230	vikas@utu.ac.in

```
SQL> commit;
```

Commit complete.

```
SQL> savepoint s1;
```

Savepoint created.

```
SQL> rollback to savepoint s1;
```

Rollback complete.

```
SQL> delete from tblstudent where enro=3;
```

1 row deleted.

CE 5.7: Data Control Language (Conti...)

```
SQL> select * from tblstudent;
```

ENRO	FNAME	LNAME	CONTACTNO	EMAILID
1	Bhavik	Sarang	9857463210	bhavik@gmail.com
2	Krishna	Patel	8965320147	krishna@gmail.com
4	Raj	Chauhan	6523987410	raj@yahoo.com
5	Vikas	Patil	7896541230	vikas@utu.ac.in

```
SQL> rollback;
```

Rollback complete.

```
SQL> select * from tblstudent;
```

ENRO	FNAME	LNAME	CONTACTNO	EMAILID
1	Bhavik	Sarang	9857463210	bhavik@gmail.com
2	Krishna	Patel	8965320147	krishna@gmail.com
3	Om	Puri	9658741230	om@yahoo.com
4	Raj	Chauhan	6523987410	raj@yahoo.com
5	Vikas	Patil	7896541230	vikas@utu.ac.in

Home Work

1. What is a transaction? [1 Mark]
2. State the significance of COMMIT and ROLLBACK. [2 Marks]
3. Discuss transaction control with an appropriate example. [5 Marks]

International Certification Question

Q1.	What does SQL stands for?
A.	Strong Question Language
B.	Structured Query Language
C.	Structured Question Language

Ans: B

International Certification Question

Q2.	Which SQL statement is used to extract data from a database?
A.	EXTRACT
B.	GET
C.	OPEN
D.	SELECT

Ans: D

International Certification Question

Q3.	Which SQL statement is used to update data in a database?
A.	MODIFY
B.	SAVE AS
C.	UPDATE
D.	EDIT

Ans: C

International Certification Question

Q4.	Which SQL statement is used to delete data from a database?
A.	DELETE
B.	DROP
C.	REMOVE
D.	THROW

Ans: A

International Certification Question

Q5.	Which SQL statement is used to insert new data in a database?
A.	INSERT INTO
B.	ADD RECORD
C.	ADD NEW
D.	INSERT NEW

Ans: A

International Certification Question

Q6.	With SQL, how do you select a column named "FirstName" from a table named "Persons"?
A.	SELECT Persons.FirstName;
B.	SELECT FirstName FROM Persons;
C.	DESC FirstName FROM Persons;
D.	EXTRACT FirstName FROM Persons;

Ans: B

International Certification Question

Q7.	How do you select all the columns from a table named "Persons" in SQL?
A.	DESC Persons;
B.	SELECT FROM Persons;
C.	SELECT * FROM Persons;
D.	DISPLAY ALL FROM Persons;

Ans: C

International Certification Question

Q8.	With SQL, how do you select all the records from a table named "Persons" where the value of the column "FirstName" is "Bhavik"?
A.	SELECT * FROM Persons WHERE FirstName<>'Bhavik';
B.	SELECT [all] FROM Persons WHERE FirstName LIKE 'Bhavik';
C.	SELECT [all] FROM Persons WHERE FirstName='Bhavik';
D.	SELECT * FROM Persons WHERE FirstName='Bhavik';

Ans: D

International Certification Question

Q9.	Which SQL statement is used to return only different values?
A.	SELECT DIFFERENT
B.	SELECT DISTINCT
C.	SELECT UNIQUE
D.	SELECT REPEAT

Ans: B

International Certification Question

Q10.	Which SQL keyword is used to sort the result-set?
A.	ORDER BY
B.	SORT BY
C.	ORDER
D.	SORT

Ans: A

International Certification Question

Q11.	With SQL, how can you return all the records from a table named "Persons" sorted descending by "FirstName"?
A.	SELECT * FROM Persons SORT BY 'FirstName' DESC;
B.	SELECT * FROM Persons ORDER BY FirstName DESC;
C.	SELECT * FROM Persons SORT 'FirstName' DESC;
D.	SELECT * FROM Persons ORDER FirstName DESC;

Ans: B

International Certification Question

Q12.	With SQL, how can you insert “Sarang” as the "LastName" in the "Persons" table?
A.	INSERT INTO Persons (LastName) VALUES (‘Sarang’);
B.	INSERT (‘Sarang’) INTO Persons (LastName);
C.	INSERT INTO Persons (‘Sarang’) INTO LastName;
D.	INSERT INTO Persons VALUES (LastName=‘Sarang’);

Ans: A

International Certification Question

Q13.	How can you change "Bhavik" into "Raj" in the "FirstName" column in the Persons table?
A.	MODIFY Persons SET FirstName='Raj' WHERE FirstName='Bhavik';
B.	UPDATE Persons SET FirstName='Bhavik' INTO FirstName='Raj';
C.	MODIFY Persons SET FirstName='Bhavik' INTO FirstName='Raj';
D.	UPDATE Persons SET FirstName='Raj' WHERE FirstName='Bhavik';

Ans: D

International Certification Question

Q14.	How can you delete the records where the "City" is "Vapi" in the Persons Table in SQL?
A.	DELETE City='Vapi' FROM Persons;
B.	DELETE FROM Persons WHERE City='Vapi';
C.	DELETE ROW City='Vapi' FROM Persons;
D.	DELETE FROM Persons City is 'Vapi';

Ans: B

International Certification Question

Q15.	Which SQL statement is used to create a table in a database?
A.	CREATE TABLE
B.	CREATE DATABASE TABLE
C.	CREATE DATABASE TAB
D.	CREATE DB

Ans: A

International Certification Question

Q16.	Which constraint is the combination of NOT NULL and UNIQUE KEY?
A.	PRIMARY KEY
B.	FOREIGN KEY
C.	CHECK
D.	Default

Ans: A

International Certification Question

Q17.	The NOT NULL constraint enforces a column to not accept null values.
A.	False
B.	True

Ans: B

International Certification Question

Q18.	Which constraint is used to link two tables?
A.	PRIMARY KEY
B.	FOREIGN KEY
C.	CHECK
D.	Default

Ans: B

International Certification Question

Q19.	Which of the following statement is correct?
A.	Alter table employee alter salary set default:2000;
B.	Alter table employee alter salary set default=2000;
C.	Alter table employee alter salary set default 2000;
D.	Alter table employee add salary set default 2000;

Ans: C

International Certification Question

Q20.	TRUNCATE statement in SQL is a.....
A.	DDL statement
B.	DML statement
C.	DCL statement
D.	DQL statement

Ans: A

Industry Interview Questions

1. What is SQL/MySQL?
2. What are tables in SQL?
3. What are the different types of statements supported by SQL?
4. How do we use DISTINCT statement? What is its use?
5. Which are different Clauses used in SQL?
6. By which, clause you can sort data in MySQL?
7. How can you retrieve top 10 records from a table?
8. What is a Unique key?
9. What is a primary key?
10. What are foreign keys?
11. What is CHECK Constraint?
12. Why do we use SQL constraints? Which constraints we can use while creating database in SQL?

Industry Interview Questions

13. How can you restrict for entering NULL data in the table?
14. Is it possible for a table to have more than one foreign key?
15. What is the difference between DELETE and TRUNCATE?
16. What is the difference between DROP and TRUNCATE?
17. What is the use of COMMIT?
18. Can we rollback our records, which is removed with the help of TRUNCATE? Justify your answer.
19. What are transaction and its controls?
20. What is the difference between SQL and MySQL?

Difference between SQL and MySQL

- SQL is a structured query language that is used for manipulating and accessing the relational database.
- MySQL itself is a relational database that uses SQL as the standard database language.

Thank You