CSE 311L(Database Management System)

LAB-Week 01(Part A)

Objectives:

After completing this lesson, you should be able to do the following:

- Create database tables
- Describe the data types that can be used when specifying column definition
- Table naming rules & Fields Datatypes

Table Naming Rules

Table names and column names:

- Must begin with a letter
- Must be 1–30 characters long
- Must contain only A–Z, a–z, 0–9, _, $, and #
- Must not duplicate the name of another object
- owned by the same user
- Must not be an Oracle server reserved word

Data Types:

Data type	Description
VARCHAR(LEN)	Variable length char. Values up to width length. Maximum width is 4000 chars.
CHAR(LEN)	Fixed length char. Values of width length. Default width is 1. Maximum length is 200.
INT	Floating point with precision of 38 significant digits.
FLOAT (Precision, Scale)	Precision represents Maximum significant digits allowed, which may not exceed 38. Scale is the number of decimal places on the right of the decimal point.

DATE	Date values in the range 1-1-4712 B.C to 31-12-4712 AD.
LONG	Variable length char. Values up to 2 GB. Only one LONG col. is allowed per table. You cannot use LONG datatype in functions, WHERE clause and sub queries.
RAW AND LONG RAW	Equivalent to VARCHAR2 and LONG respectively, used for storing digital sound or graphics images.
CLOB, BLOB, NCLOB	Used to store large char and binary Objects .Each can accommodate up to 4 GB.
BFILE	Stores a pointer to an external file.
TIMESTAMP	It is an extension of Date data type. It stores year, month, day, hour, minute and second values

CREATE TABLE Statement

CREATE TABLE

Specifies a new base relation by giving it a name, and specifying each of its attributes and their data types

Syntax of CREATE Command:

CREATE TABLE <*table name***>** (<Attribute *A*1> <Data Type *D*1> [< Constarints>], <Attribute *A*2> <Data Type *D*2> [< Constarints>],

…….
<Attribute *A*n> <Data Type *D*n> [< Constarints>],
[<integrity-constraint1>, <integrity-constraint k>]);

Example: CREATE TABLE DEPARTMENT (
DNAME VARCHAR(10) NOT NULL,
DNUMBER INTEGER NOT NULL,
MGRSSN CHAR(9), MGRSTARTDATE CHAR(9));

- A constraint NOT NULL may be specified on an attribute

Integrity Constraints Description

An integrity constraint is a declarative method of defining a rule for a column of a table. Oracle supports the following integrity constraints:

- NOTNULL constraints for the rules associated with nulls in a column
- UNIQUE key constraints for the rule associated with unique column values
- PRIMARYKEY constraints for the rule associated with primary identification values
- FOREIGNKEY constraints for the rules associated with referential integrity. Oracle supports the use of FOREIGNKEY integrity constraints to define the referential integrity actions, including:
  - Update & delete no action
  - Delete CASCADE
  - Delete SET NULL
- CHECK constraints for complex integrity rules

Referential Integrity Constraints

Different tables in a relational database can be related by common columns, and the rules that govern the relationship of the columns must be maintained. Referential integrity rules guarantee that these relationships are preserved.
The following terms are associated with referential integrity constraints.

Term	Definition
Foreign key	The column or set of columns included in the definition of the referential integrity constraint that reference a referenced key
Referenced key	The unique key or primary key of

	the same or different table that is referenced by a foreign key.
Dependent or child table	The table that includes the foreign key. Therefore, it is the table that is dependent on the values present in the referenced unique or primary key
Referenced or parent table	The table that is referenced by the child table's foreign key. It is this table's referenced key that determines whether specific inserts or updates are allowed in the child table

Specifying the unique, primary key attributes, secondary keys, and referential integrity constraints (foreign keys).

Example:

```
CREATE TABLE DEPT (
        DNAME VARCHAR(10) NOT NULL,
        DNUMBER INTEGER NOT NULL,
        MGRSSN CHAR(9),
        MGRSTARTDATE CHAR(9),
        PRIMARY KEY (DNUMBER),
        UNIQUE (DNAME),
        FOREIGN KEY (MGRSSN) REFERENCES EMPLOYEE(SSN));
```

We can specify RESTRICT, CASCADE, SET NULL or SET DEFAULT on referential integrity constraints (foreign keys)

Example:

```
CREATE TABLE DEPT (
        DNAME VARCHAR(10) NOT NULL,
        DNUMBER INTEGER NOT NULL,
        MGRSSN CHAR(9),
        MGRSTARTDATE CHAR(9),
        PRIMARY KEY (DNUMBER),
        UNIQUE (DNAME),
```

FOREIGN KEY (MGRSSN) REFERENCES EMPLOYEE(SSN)
ON DELETE SET NULL ON UPDATE CASCADE);

Activity 01:

Write SQL statement for create the 'Emps' table:
(You will be given demo of the code. Just follow it and ask questions, if there is any)

Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

Syntax Column_Constraint: [constraint name]{[NOT] NULL| {UNIQUE | PRIMARY KEY}|
REFERENCES table [(column)]| CHECK (condition)}

Ex:SQL>create table subjects(subdesc varchar(20) constraint const_name NOT NULL..);

Ex2:SQL>create table subjects(subdesc varchar(10) constraint subject_pk PRIMARY KEY);

Syntax Table_Constraint: [constraint name]{{UNIUQE | PRIMARY KEY} (**Col, [, col]…)**|

FOREIGN KEY **(Col, [, Col}…)** REFERENCES **table [(Col {, Col]…)**| CHECK (condition)}

Ex1: SQL>Create table studm**arks (…,constraint studmarks_chk** Check(stdate<=enddate));

Ex2: syntax for table constraint

SQL> Create table studmars (sno number (5), constraint studmarks_fk FOREIGN KEY (sno) references students (sno));

DROP TABLE

- Used to remove a relation (base table) and its definition.
- The relation can no longer be used in queries, updates, or any other commands since its description no longer exists

  Example: DROP TABLE DEPTS;

ALTER TABLE:

- Used to add an attribute to/from one of the base relations drop constraint -- The new attribute will have NULLs in all the tuples of the relation right after the command is executed; hence, the NOT NULL constraint is *not allowed* for such an attribute.

  Example: ALTER TABLE EMPS ADD JOB VARCHAR(12);
- The database users must still enter a value for the new attribute JOB for each EMPLOYEE tuple. This can be done using the UPDATE command.

DROP A COLUMN (AN ATTRIBUTE)

- ALTER TABLE COMPANY.EMPLOYEE DROP ADDRESS CASCADE; All constraints and views that reference the column are dropped automatically, along with the column. ALTER TABLE COMPANY.EMPLOYEE DROP ADDRESS RESTRICT; Successful if no views or constraints reference the column. ALTER TABLE COMPANY.DEPARTMENT ALTER MGRSSN DROP DEFAULT;

CSE 311L(Database Management System)

LAB-Week 01(Part B)

Objectives:

After completing this lesson, you should be able to do the following:

- Insert rows into the created table
- Create Department Table
- Execute a basic SELECT statement

The `INSERT` Statement Syntax

```
INSERT INTO table [(column [, column...])]
VALUES (value [, value...]);
```

Activity 01:

Write SQL statement for create the 'Depts' table:

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

Activity 02:

Write SQL statement for INSERT two employees' data into the **Emps** table you create earlier.

Activity 03:

Write SQL statement for INSERT two Departments' data into the **Depts** table you just created.