****

USMAN INSTITUTE OF TECHNOLOGY

Department of Computer Science

**CS311 Database Management Systems**

**Fall 2019**

**Lab # 5 :** Creating and managing tables

**Objective:**

.

**Name of Student:**

**A**

**Roll No: Section:**

7-Nov-2019

**Date of Experiment:**

**Date Submitted:**

Marks Obtained / Remarks: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**THEORY**

**Database Objects: A**n Oracle database can contain multiple data structures. Each structure should be outlined in the database design so that it can be created during the build stage of database development.

* **Table**: Stores data
* **View**: Subset of data from one or more tables
* **Sequence**: Generates primary key values
* **Index**: Improves the performance of some queries
* **Synonym**: Gives alternative names to objects

**Oracle Table Structures**

* Tables can be created at any time, even while users are using the database.
* We do not need to specify the size of any table. The size is ultimately defined by the amount of space allocated to the database as a whole.
* Table structure can be modified.

**Naming Conventions**

* Name database tables and columns according to the standard rules for naming any Oracle database object.
* Table names and column names must begin with a letter and can be 1-30 characters long.
* Names must contain only the characters A-Z, a-z, 0-9, \_*(underscore*), $, and # (legal characters, but their use is discouraged).
* Names must not duplicate the name of another object owned by the same Oracle Server user.
* Names must not be an Oracle Server reserved word.

**Creating and Altering Tables**

**The CREATE TABLE statement**

To create a table, a user must have the CREATE TABLE privilege and a storage area in which to create objects. The database administrator uses Data Control Language (DCL) statements, covered in a later session, to grant privileges to users.

The syntax is as follows:-

|  |
| --- |
| CREATE TABLE [*schema* .] *table name*  (*column datatype* [DEFAULT *expr*] [,  …..  …]); |

**Referencing another user’s tables**

A *schema* is a collection of objects. *Schema objects* are the logical structures that directly refer to the data in a database. Schema objects include tables, views, synonyms, sequences, stored procedures, indexes, clusters, and database links.

If a table does not belong to the user, the owner’s name must be prefixed to the table.

**The DEFAULT option**

A column can be given a default value by using the DEFAULT option. This option prevents null values from entering the columns if a row is inserted without a value for the column. The default value can be a literal, an expression, or a SQL function, such as SYSDATE and

USER, but the value cannot be the name of another column or a pseudocolumn, such as NEXTVAL, or CURRVAL. The default expression must match the datatype of the column.

For example,

… hiredate DATE DEFAULT SYSDATE, …

**Example**

The following example creates the DEPT table mentioned in the lab. session 01.

|  |
| --- |
| CREATE TABLE dept (  deptno NUMBER(2),  dname VARCHAR2(14),  loc VARCHAR2(13)  ); |

Since creating a table is a DDL statement, an automatic commit takes place when this statement is executed.

In order to confirm the creation of the table, issue the DESCRIBE command as discussed in lab session 01.

DESCRIBE DEPT

**SQL Data Types**

|  |  |
| --- | --- |
| **Datatype** | **Description** |
| VARCHAR2(*size*) | Variable-length character data (A maximum *size* must be specified. Default and minimum *size* is 1; maximum *size* is 4000) |
| CHAR(*size*) | Fixed-length character data of length *size* bytes (Default and minimum *size* is 1; maximum *size* is 2000) |
| NUMBER(*p*, *s*) | Number having precision p and scale s (The precision is the total number of decimal digits and the scale is the number of digits to the right of the decimal point. The precision can range from 1 to 38 and the scale can range from -84 to 127.) |
| DATE | Date and time values between January 1, 4712 B.C. and December 31, 9999 A.D. |
| LONG | Variable length character data up to 2 gigabytes |
| **Datatype** | **Description** |
| CLOB | Single-byte character data up to 2 gigabytes |
| RAW(*size*) | Raw binary data of length *size* (A maximum size must be specified. Maximum *size* is 2000.) |
| LONG RAW | Raw binary data of variable length up to 2 gigabytes |
| BLOB | Binary data up to 4 gigabytes |
| BFILE | Binary data stored in an external file; up to 4 gigabytes |

**Table 5.1**

CLOB, BLOB and BFILE are the large object data types and can store blocks of unstructured data (such as text, graphics images, video clips and sound wave forms up to 4 gigabytes in size.) LOBs also support random access to data.

**Creating a table by using a Subquery**

The following example creates a table, DEPT30, that contains details of all employees working in department 30

|  |
| --- |
| CREATE TABLE dept30  AS SELECT empno, ename, sal \* 12 ANNUALSAL, hiredate  FROM emp  WHERE deptno = 30; |

**Altering table structure**

The ALTER TABLE statement is used to

* Add a new column
* Modify an existing column
* Define a default value for the new column

The following example **adds a new column** to the above newly created DEPT30 table: -

|  |
| --- |
| ALTER TABLE dept30  ADD (job VARCHAR2(9)); |

To **modify an existing column**, use

|  |
| --- |
| ALTER TABLE dept30  MODIFY (ename VARCHAR2(15)); |

**Dropping a Table**

The DROP TABLE statement removes the definition of an Oracle table. The database loses all the data in the table and all the indexes associated with it.

The DROP TABLE statement, once executed, is irreversible. The Oracle Server does not question the action when the statement is issued and the table is immediately dropped. All DDL statements issue a commit, therefore, making the transaction permanent.

To drop the table DEPT30,

|  |
| --- |
| DROP TABLE DEPT30; |

**Changing the name of an object**

To change the name of a table, view, sequence, or synonym, execute the RENAME statement:-

|  |
| --- |
| RENAME dept TO department; |

**What are constraints?**

The Oracle Server uses constraints to prevent invalid data entry into tables.

Constraints are used for the following purposes:-

* Enforce rules at the table level whenever a row is inserted, updated, or deleted from that table. The constraint must be satisfied for the operation to succeed.
* Prevent the deletion of a table if there are dependencies from other tables.  Provide rules for Oracle tools, such as Oracle Developer.

The following constraint types valid in Oracle: -

|  |  |
| --- | --- |
| **Constraint** | **Description** |
| NOT NULL | Specifies that this column may not contain a null value |
| UNIQUE | Specifies a column or combination of columns whose values must be unique for all rows in the table |
| PRIMARY KEY | Uniquely identifies each row of the table |
| FOREIGN KEY | Establishes and enforces a foreign key relationship between the column and a column of the referenced table |
| CHECK | Specifies a condition that must be true |

**Table 5.2**

**Constraint Guidelines**

* All constraints are stored in the data dictionary
* Name a constraint or the Oracle server will generate a name by using the SYS\_Cn format
* Create a constraint
* At the same time as the table is created.

The EMP table is being created specifying various constraints:-

|  |
| --- |
| CREATE TABLE DEPT (  DEPTNO NUMBER(2) constraint DEPT\_DEPTNO\_PK PRIMARY KEY,  DNAME VARCHAR2(14),  LOC VARCHAR2(13),  CONSTRAINT DEPT\_DNAME\_UK NIQUE(DNAME)  ); |

|  |
| --- |
| CREATE TABLE EMP (  EMPNO NUMBER (4) CONSTRAINT EMP\_EMPNO\_PK PRIMARY KEY,  ENAME VARCHAR2(10) NOT NULL,  JOB VARCHAR2(9),  MGR NUMBER (4),  HIREDATE DATE DEFAULT SYSDATE,  SAL NUMBER(7, 2), COMM NUMBER(7, 2),  DEPTNO NUMBER(7, 2) NOT NULL,  CONSTRAINT EMP\_DEPTNO\_CK CHECK (DEPTNO BETWEEN 1 AND 50),  CONSTRAINT EMP\_DEPTNO\_FK FOREIGN KEY (DEPTNO) REFERENCES DEPT(DEPTNO)  ); |

Composite primary keys are defined at the table level.

**EXERCISES**

Consider the following schema, in the form of normalized relations, to represent information about *employees*, *grades*, *training* and *projects* in an organization.

|  |  |
| --- | --- |
| **EMPLOYEE** | **GRADE** |
| Empno (eg 6712)  Name  Designation (e.g. *Database Developer*)  Qualification  Join date | Designation  Grade (1-20)  TotalPosts  PostsAvailable (<= TotalPosts) |
| **PROJECT** | **TRAINING** |
| PID (eg P812)  Title  Client  Duration (in weeks)  Status (New, In Progress, Complete) | Tcode (eg T902)  Title  StartDate  EndDate |
| **EMP\_PROJECT** | **EMP\_TRAINING** |
| Empno  PID  Performance (Excellent, Good, Fair, Bad, Poor) | Empno  Tcode  Attendance (%) |

1. Develop a script file **EMPLOYEE.SQL** to create tables for the above schema. Implement all necessary *integrity constraints* including primary and foreign keys. (NOTE: All ***check*** constraints should be at table level)

1. Write SQL statements to add

* *Gender* column to **EMP** table. The only possible values are *Male* and *Female*.
* *Instructor\_Name* column to **TRAINING** table.
* *Salary* column to **GRADE** table.

1. What is *database schema*? What are the different objects included in it?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_