LO4 and LO5 – Table Creation with DDL

DDL – Data Definition Language (LO4 and LO5)

1. We are changing the structure of the database.
2. Commands which create new database objects, modify and delete existing database objects
3. When you execute a DDL command, the database immediately changes
4. Ex: ALTER TABLE, CREATE TABLE, DROP TABLE, RENAME

DML – Data Manipulation Language (LO6)

1. We are manipulating the contents of the database.
2. Commands insert, update, delete, and view database objects
3. When you execute a DML command, you must explicitly save the command to make the new data values commit
4. Ex: INSERT, UPDATE, DELETE, SELECT

SQL statements are embedded in scripting languages or programming languages. DDL and DML are subsets of SQL.

Graphical user interface

Description automatically generated

# Data Definition Language

CREATE TABLE statement

* Syntax:  
    
  CREATE TABLE TableName(  
   columnName datatype[constraint]  
   {, columnName datatype [constraint]}…  
   {, [constraints]}…

);

* Each column definition has 3 parts

1. Name of the column
2. Datatype of the column
3. Column constraints (optional)

DROP TABLE statement

* Syntax:

DROP TABLE[schema.]tableName[CASCADE CONSTRAINTS]

* CASCADE CONSTRAINTS allows a parent table to be dropped even if a child table exists

# Oracle Data Types

* A data type of a column specifies what kind of information the column will hold (characters, numbers, dates, etc…) so that the system will know how the data is to be physically stored and how it can be manipulated.
* It is very important to pick the correct datatype because it is usually difficult to change it later. You would need to use a datatype conversion function
* Oracle does not use INT or TINYINT – instead we use NUMBER with an appropriate size. INTERGER and SMALLINT are the same as NUMBER(38).
* Oracle does not reasonably support DECIMAL (it lets you use DECIMAL, but ends up treating it as if it were a NUMBER(38) – with no decimal portion). Instead, use NUMBER(size, #decimals) or DECIMAL(size, #decimal). 999.99 -> NUMBER(5,2)
* Oracle stores date and time values in columns using the DATE datatype. DATE’s default format uses DD-MON-YY, but there is a time portion being maintained.

# Number Data Types (NUMBER)

* Stores negative, positive, fixed and floating point number between 10-130 and 10125 with a precision up to 38 decimal places.
* columnName NUMBER [([precision,][scale])]
  + precision stores the **total number of digits**
  + scale is the number of digits on the right side of the decimal point
* Subtypes:
  + Integer numbers
    - columnName NUMBER(precision)
    - NUMBER(3) – three digits with no decimal
    - Decimals are rounded, and an error is generated if the value is too large
  + Fixed-point numbers
    - columnName NUMBER(6,2)
    - fixed number of decimal places
    - In this example, 6 digits, 4 on the left, and 2 on the right
  + Floating-point numbers
    - columnName NUMBER
    - Decimal can appear anywhere from before the first digit to after the last digit, or not at all -> 1.3, 1.0000004, .0004

# Character Datatypes

* CHAR
  + columnName CHAR [(max size)]
  + stores fixed-length character data up to a maximum of 2000 characters
  + max size is optional - default is size 1
  + Trailing spaces are added to pad up to the max size
* VARCHAR2
  + columnName VARCHAR2(maxSize)
  + Always uses VARCHAR2 and not VARCHAR
  + Stores variable length character data up to 4000 characters
  + **Must specify a size**
* NVARCHAR2 and NCHAR
  + Only allows ASCII character set. Similar to VARCHAR2 and CHAR.

# Date and Time Datatypes

* Datatypes that store date and time values include **datetime** data subtypes, which store actual date and time values, and the **interval** data subtypes, which store an elapsed time interval between two date/time values. The main datetime subtypes are DATE and TIMESTAMP. The interval subtypes include INTERVAL YEAR TO MONTH and INTERVAL DAY TO SECOND.
  + DATE
    - columnName DATE
    - Stores dates from December 31, 4712 BC to December 31, 4712 AD.
    - Stores the century, year, month, day, hour, and second
    - Default format is DD-MON-YY – if no time is entered with the date the default is 12:00:00 AM
    - Default time format is HH:MI:SS AM **– using the 12 hour clock**
    - If no date is entered with a time the default is the first day of the current month
  + TIMESTAMP
    - columnName TIMESTAMP (fractional seconds precision)
    - stores the date values similar to DATE datatype, except it also stores fractional seconds.
    - The default is 6 decimal places for fractional seconds precision

# Large Object (LOB) Datatypes

* columnName LOB\_data\_type
* Used to store binary data, such as digitized sounds or images or references to binary files from a word processor or spreadsheet.
* There are 4 LOB datatypes:

1. BLOB – Binary LOB stores up to 4GB of data in the database
2. BFILE – Binary files stores a reference to a binary file located outside the database in a file maintained by the OS
3. CLOB – Character LOB stores up to 4GB of character data in the database
4. NCLOB – Character LOB that supports 2-byte character codes, stores in the database – up to 4GB

# Constraints

* Constraints are rules that restrict the data values that you can enter into a column in a database table.
* There are two types of constraints:
  + Integrity constraints: which define primary keys and foreign keys
  + Value constraints: which defines specific data values or data ranges that must be inserted into columns and whether the values must be unique, or not null.
* There are two levels of constraints
  + Table constraints: restricts the data value with respect to all other values in the table.
    - Ex: Primary key constraint – specifies that a column value must be unique and cannot appear in more than one record.
  + Column constraints: limit the value that can be placed in a specific column, regardless of values that exists in other records.
* You can place constraint definitions at the end of the CREATE TABLE command, after you declare all of the columns.
* OR you can place each constraint definition within the column definition.
* Each constraint must have a unique constraint name. You can define your own or omit them and Oracle will name your constraint. Max of 30 characters.

|  |  |
| --- | --- |
| **Constraint Type** | **Constraint ID Abbreviation** |
| Primary Key | PK |
| Foreign Key | FK |
| Check | CC |
| Null / Not Null | NN |
| Unique | UK |

Use TableName\_columnName\_constraintIDAbbreviation

Ex: Project\_projectID\_pk

Project\_projectID\_PK

Types of constraints

* **Integrity constraints**
* Defines primary key columns and specifies foreign keys corresponding table and column references.
* Primary Keys
* Syntax:
  + Within a column
    - columnName DataType CONSTRAINT constraintName\_PK PRIMARY KEY
  + At the end of the CREATE TABLE
    - CONSTRAINT constraintName\_PK PRIMARY KEY (columnName)
* Foreign Keys
* Syntax:
  + Within a column
    - columnName DataType CONSTRAINT constraintName\_FK REFERENCES ParentTable(columnNameInParent)
  + At the end of the CREATE TABLE
    - CONSTRAINT constraintName\_FK FOREIGN KEY (columnName) REFERENCES ParentTable(columnNameInParent)
  + Note: Before you can create a Foreign Key constraint the parent table must already exist.
* Composite Key
* You will need to list all of the columns involved.
* Syntax (only creatable at the end of the CREATE TABLE statement):
  + CONSTRAINT constraintName\_PK PRIMARY KEY (columnName1, columnName2, …)
* **Value Constraints**
* Column-level constraints that restrict the data values that users can enter into a given column
* Check, Not Null, and Default
* Check constraints
  + Specify that a column value must be a specific value or fall within a range of values.
  + Make sure you specify all conditions because once the table is created and populated it is difficult to modify the constraint.
  + DBMS must be able to evaluate each check condition to either true or false. You can combine expressions using logical operators AND and OR.
  + Syntax:
    - columnName DataType CONSTRAINT constraintName\_CC CHECK (expression)
  + Examples:
    - CONSTRAINT student\_gender\_CC CHECK ((gender=’M’) OR (gender=’F’) OR (gender=’m’) OR (gender=’f’))
    - CONSTRAINT course\_credits\_CC CHECK ((credits>0) AND (credits<=12))
* NOT NULL Constraints
  + Specify whether the user MUST enter a value for a specific field or whether the value can be NULL.
  + NULL is the default for fields except the Primary Key
  + Primary Keys are automatically set to NOT NULL.
  + Syntax:
    - columnName DataType CONSTRAINT constraintName\_NN NOT NULL
* Default Constraints
  + Specifies that a particular column has a default value that the DBMS automatically inserts for every record, unless the user specifies an alternate value.
  + These must be created in the column definition and do not begin with the CONSTRAINT keyword
  + Syntax:
    - columnName DataType DEFAULT defaultValue
* Unique Constraints
  + Specifies that a column must have a unique value for every record
  + All primary keys are automatically assigned a unique constraint
  + Like the definition of the primary key constraint except NULL values ARE allowed.
  + Syntax:
    - CONSTRAINT constraintName\_UK UNIQUE (columnName)

# System Tables

We can retrieve information about a variety of database objects using different data dictionary views. These views simply contain data (structure and contents) about our data; this Metadata can be queried, and the result is a “view”.

|  |  |
| --- | --- |
| Object Name | Object Type |
| User\_Objects | All database objects |
| User\_Tables | Database tables |
| User\_Indexes | Table indexes create to improve query retrieval performance |
| User\_Views | Database views |
| User\_Constraints | Table constraints |
| User\_CONS\_Columns | Table columns that have constraints |
| User\_IND\_Columns | Tables columns that have indexes |
| User\_Tab\_Columns | All table columns. |

# Dropping Tables

Syntax: DROP TABLE tableName [CASCADE CONSTRAINTS]

* When a table is dropped, the system will remove all data from the table and delete the table and any associated indexes.
* If a table is the “parent” table (meaning another table uses this one for foreign keys), then the table cannot be dropped by itself. You either have to remove the foreign key constraint (DROP CONSTRAINT, or drop the child table), OR use CASCADE CONSTRAINTS to force the drop.

## Unrestricted Actions and Restricted Actions

Various actions involving the database schema are classified as restricted or unrestricted.

* Modifying and Deleting Database Tables

1. Unrestricted Actions
   * + Specifics on database tables you can always modify:
       - Renaming a table
       - Adding new fields
       - Deleting fields
       - Increasing the maximum size of a field
       - Delete constraints
2. Restricted Action

|  |  |
| --- | --- |
| **Restricted Action** | **Restriction** |
| Deleting a Table | Allowed only if the table does not contain any fields that other tables reference as foreign keys. |
| Changing an existing field’s data type | Allowed only if existing data in the field is compatible with the new data type. |
| Decreasing the width of an existing field. IE: Going from NVARCHAR2(32) NVARCHAR2(16) | Allowed only if existing field values are null |
| Add a primary key constraint to an existing field | Allowed only if current field values are UNIQUE and NOT NULL |
| Add a foreign key constraint | Allowed only if current field values are NULL or existing in the referenced table. |
| Adding a UNIQUE constraint to a field | Allowed only if current field values are already unique relative to each other. |
| Adding a CHECK constraint | Allowed, but only applies to new values inserted. |
| Change a field’s DEFAULT value | Allowed, but only applies to new values inserted. |

## Renaming Existing Tables

Syntax: RENAME OldTableName TO NewTableName

* When renaming a table, the DBMS automatically transfers to the new table integrity constraints, indexes, and privileges that referenced the old table name.
* However, when you rename a table, objects that referenced the old table, such as views and stored procedures and functions, become invalid.

## Alter Table - Making Changes to Existing Tables

* After a table has been created, its structure, properties, or constraints can be changed using the ALTER statement.
* Altering an empty table usually poses no difficulties. Altering a populated table may require ninja DBA skills to fix the data before the alteration can occur.

### Adding Fields to an Existing Table

Unrestricted action

Syntax: ALTER TABLE TableName ADD (columnName datatype [constraints]);

### Deleting Fields from Existing Table

When you delete a column (DROP), it also drops all associated constraints

Syntax: ALTER TABLE TableName DROP COLUMN columnName

### Adding and Deleting Constraints Using Alter

* Add a new constraint is a restricted action. It is difficult to add new constraints to tables that already contain data, because usually the data values do not conform to the new constraint.
* Deleting constraints is unrestricted.
* Add Syntax:
  + ALTER TABLE TableName ADD CONSTRAINT constraintName constraintDefinition
* Delete Syntax:
  + ALTER TABLE TableName DROP CONSTRAINT constraintName

### Modifying Existing Fields Using Alter Table

* Can modify only if existing values are compatible with the new data type.
* Making the field max size larger is unrestricted but making it smaller is restricted.
* Syntax:
  + ALTER TABLE TableName MODIFY (columnName new\_data\_definition)