LO4 and LO5 – Table Creation with DDL

DDL – Data Definition Language (LO4 and LO5)

1. We are changing the structure of the database. –making tables, or changing,deleting tables
2. Commands that create new Db objects and modify existing DB objects
3. Examples: CREATE, ALTER, DROP, RENAME
4. When you execute a DDL command, the DB automatically changes(commit occurs)

DML – Data Manipulation Language (LO6)

1. We are manipulating the contents of the database.
2. Commands that insert, update, delete and view the DB objects’ content (data)
3. EX: INSERT, UPDATE, DELETE and SELECT
4. When you execute a DML command, you have to explicitly save the changes (commit). A rollback can be issued to undo the changes(if you don’t commit)

# Data Definition Language

CREATE TABLE statement

[] = optional

* Syntax:  
  CREATE TABLE tableName(

columnName datatype [constraints],

columnName datatype [constraints]…….

[end of table constraints]

);

* Each column definition has 3 parts:
  + Name of the column-required
  + Datatype of the column – required
  + Column Constraints(optional)

DROP TABLE statement

* Syntax:  
  DDROP TABLE [schema.]tableName [CASCADE CONSTRAINTS];
* Cascacde constraints allows a parent table to drop even if a child table with records exists.

# Oracle Data Types

-Tell the system what is being stored in the system so it knows how to handle it

-A data type of a column specifies what kind of info that the column will hold (characters, numbers, dates, etc) so 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 data type because it is usually difficult to change it later. You may need to create a conversion function or program if you already have data in.
* Oracle does not uise INT or TINYINT, use NUMBER instead. INTEGER will end up a NUMBER(38). Use NUMBER instead
* Likewise, DECIMAL is supported, but is treated as a NUMBER(38).

# Number Data Types

* Stores positive, negative, fixed, and floating point numbers between 10^-38 and 10^125 with a precision of up to 38 decimals places.
* columnName NUMBER[(precision[,scale])] [] = optional
* subtypes based on parameters used:
  + Integers: columnName NUMBER(precision)
    - Ex: NUMBER(3) would be 3 digits with no decimal range of -999 to 999
    - Decimals numbers are rounded, error generated if larger then allowed value
  + Fixed point Numbers: columnNmae NUMBER (precision, scale)
    - Precision is number of digits, scale is number of decimal places
    - EX:NUMBER(6,2) would have 6 digits with 4 on left of decimal, 2 on the right -9999.99 -> 9999.99. The decimal place is fixed, will always have 4 digits on the left, 2 on right
  + Floating point number:columnName NUMBER
    - Decimal can appear anywhere from before the firs digit to after the last

# Character Datatypes

* CHAR
  + Fixed length character/string
  + columnName CHAR[(maxsize)]
  + Stores fixed length character data up to a maximum of 200 characters
  + Max size is optional, with default of 1
  + Trailing spaces are added to pad up to the max size
  + So CHAR(4), and ‘aa’ is entered it would appear as ‘aa\_ \_’
* VARCHAR2
  + Variable length string
  + ColumnNameVARCHAR2(maxSize)
  + Stores variable length character up to 4000 charactres
  + MaxSize is required
* NCHAR and NVARCHAR2
  + NCHAR/NVARCHAR2 only allow asci characters, similar to CHAR and VARCHAR2

# 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, minute, and second
    - Degault format is DD-Mon-YY
    - Defauilt time format is HH:Min:SS AM – using the 12 hour clock
  + TIMESTAMP
    - columnName TIMESTAMP (frational secods precision)
    - stores the date values similar to DATE datatype, except it also stores fractional seconds
    - the default is 6 decimals places for fractional seconds precision
  + INTERVAL
    - <http://www.mysqltutorial.org/mysql-interval/>

# 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 store 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 DB
  + 4. NCLOB – Character LOB that supports 2-byte character codes, stores in the DB, 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 2 types of contraintsL
  + Integrity constraints: which define primary and foreign keys
  + Value constraints: Which define specific data calues or date ranges that must be followed, or whether a column can be null. Etc.
* There are 2 levels of constraints:
  + Table constraints: Table xonstraints restrict the data which respect to all other values in the table
    - Ex: Primary Key constraint – specifies that a column must be unique and cannot appear in more than one record
  + Column Constraints: limit the value that can be places in a specific column regardless of values that exist 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 oen or omit them and Oracle will name your constraint, MAX 30 charaters

|  |  |
| --- | --- |
| **Constraint Type** | **Constraint ID Abbreviation** |
| Primary | Pk |
| Foreign | Fn |
| Check | Cc |
| Null / Not Null | Cc |
| Unique | uk |

Use TableName\_columnName\_constraintIDAbbreviation

Ex: 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 CREATEA TABLE

CONSTRAINT constraintName\_pk PRIMARY KEY (columnName)

* + - Foreign Keys
      * Syntax:
        + Within a column

colomnName datatype CONSTRAINT constraintName\_fk REFERENCES parentTable(columnNameInParent)

* + - * + At the end of the CREATE TABLE

CONSTRAINT constraintName\_fk FOREIGN KEY (columnNameInTable) 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 (column-Name, columnName2, columnName3, …)
      * Defining composite foreign keys works the same way
  + Value Constraints
    - Column-level constraints that restrict the data values that users can enter into a given column
    - Check, Not Null , and default
      * Specify that a column value must be a specific value or fall within a range of values.
      * Make sure you specify all condition 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:
        + CONSTRAINT constraintName\_cc CHECK(expression)
      * Examples:
        + CONSTRAINT course\_grade\_cc CHECK ((grade > 0 ) AND (grade <= 100))
    - NOT NULL Constraints
      * Specify whether the user MUST enter a value for a specific field or whether the value can be NULL
      * Syntax (Must be in the column definition)
        + 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 – many systems do not consider default a constraint, instead it is a field property.
      * Syntax:
        + columnName datatype DEFAULT defaultValue
    - UniqueConstraints
      * Specifies that a column must have a unique value for every record
      * All primary keys are automatically assigned a unique constrain
      * Null values are allowed
      * Syntax:
        + columnName datatype CONSTRAINT constraintName\_uk UNIQUE
        + 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 scheme objects |
| User\_tables | Schema tables |
| User\_indexes | Indexes created for keys and improving query performance |
| User\_views | Schema Views |
| User\_Sequences | Sequences used to create auto numbers |
| User\_constraints | Constraints placed on columns |
| Cons\_columns | Columns that have constraints |
| Tab\_columns | All schema columns |

# Dropping Tables

Syntax: not sure why this is here but its DROP TABLE tableName

## Unrestricted Actions and Restricted Actions

* Modifying and Deleting Database Tables

1. Unrestricted Actions
   * + Renaming a table
     + Adding new fields
     + Deleting non key fields
     + Making a field larger
     + Deleting constraints
2. Restricted Action

|  |  |
| --- | --- |
| **Restricted Action** | **Restriction** |
| Delete a table from user schema | Allowed only if the table does not contain any fields that other tables reference as foreign keys |
| Changing an existing fields data type | Only if existing data in the field in compatible with the new data type |
| Decreasing the size of a field | Only if all existing data is null or the field has no values |
| Adding a foreign key constraint to an existing field | Allows only if the current field values are NULL or existing in the referenced table |
| Adding a primary key constraint to an existing field | Adding a unique constraint to an existing field |
| Adding a unique constraint to an existing field | Only if existing values are already unique |
| \*\*Adding a check constraint | Unrestricted – but constraint applies only to new changes in the table |
| \*\*Changing a fields Default View | Unrestricted – applies only to rows added after the change |

## Renaming Existing Tables

Syntax: RENAME oldTableName TO newTableName

* The DBMS automatically transferes to the new name the integrity constraint, indexes, and privileges that references the old name.
* Objects such as functions, procedures, views, etc. (as well as external code or reports) are not automatically updated – this can cause problems

## 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…,);

* Basically a column definition inside the ADD brackets

### Deleting Fields from Existing Table

In many cases the Db will automatically delete the associated constraints on the column, but it depend on the DBMS

Syntax: ALTER TABLE tableName DROP COLUMN columnName;

* A column can be renamed with ALTER TABLE tableName RENAME COLUMN old TO newName

### Adding and Deleting Constraints Using Alter

* Adding a new constraint is restricted(see above)
* Deleting a constraint is unrestricted
* Syntax:
  + ALTER TABLE tableName ADD CONSTRAINT constraintName constraintDefiinition
  + constraintDefinition would be the same form as adding at the end of the CREATEATABLE
  + 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 a field size larger is unrestricted, but making it smaller is restricted
* ALTER TABLE tableName MODIFY (columnname new\_data\_declaration)