Database Lab

LECTURE 1

DATA DEFINITION LANGUAGE (DDL)

Database Languages

- **Data definition language (DDL)**
  - defines data types and the relationships among them
- **Data manipulation language (DML)**
  - performs tasks such as inserting, updating, or deleting data occurrences
- **Query language**
  - allows searching for information and computing derived information

SQl Statements

Select	Data retrieval
INSERT UPDATE DELETE	Data manipulation language (DML)
CREATE ALTER DROP RENAME TRUNCATE	Data Definition language (DLL)
COMMIT ROLLBACK SAVEPOINT	Transaction Control

Tables in the Oracle Database

1. **User Tables:**
   - Are a collection of tables created and maintained by the user Contain user information

2. **Data Dictionary:**
   - Is a collection of tables created and maintained by the Oracle Server
   - Contain database information

Create Table

The simplified syntax for the CREATE TABLE statement is as follows:

CREATE TABLE *table_name* **(**

column_name type [DEFAULT *default_exp* CONSTRAINT *constraint_def*]

[, *column_name type* [DEFAULT *default_exp* CONSTRAINT *constraint_def*]...]

) ;

- ***table_name*** specifies the name you assign to the table.
- ***column_name*** specifies the name you assign to a column.
- ***type*** specifies the type of a column.
- ***constraint_def*** specifies the definition of a constraint on a column.
- ***default_exp*** specifies the expression used to assign a default value to a column.

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
VARCHAR2(*size*)	Variable-length character data
CHAR(*size*)	Fixed-length character data
NUMBER(*p*,*s*)	Variable-length numeric data
DATE	Date and time values
BLOB	Binary data up to 4 gigabytes

Number Data type

- The syntax of the NUMBER data type:

  NUMBER[(precision [, scale])]

- The Oracle NUMBER data type has precision and scale.
  - The precision is the number of digits in a number. It ranges from 1 to 38.
  - The scale is the number of digits to the right of the decimal point in a number.

The DEFAULT Option

- Specify a default value for a column during an insert.
- Literal values, expressions, or SQL functions are legal values.
- The default data type must match the column data type.

Example :

```
….. Salary number (6,2) default 0 …..
```

What are Constraints?

- Constraints enforce rules at the table level.
- The following constraint types are valid:
  - NOT NULL
  - UNIQUE
  - PRIMARY KEY
  - FOREIGN KEY
  - CHECK

Constraint Guidelines

- Name a constraint or the Oracle server generates a name by using the SYS_C*n* format.
- Create a constraint either:
  - At the same time as the table is created, or
  - After the table has been created
- Define a constraint at the column or table level.
- View a constraint in the data dictionary.

```
SELECT * FROM USER_CONSTRAINTS;
```

Defining Constraints

- **Column constraint level**

```
column [CONSTRAINT constraint_name] constraint_type,
```

- **Table constraint level**

```
column,...
[CONSTRAINT constraint_name] constraint_type
(column, ...),
```

- In the syntax:
  - ***constraint_name*** is the name of the constraint
  - ***constraint_type*** is the type of the constraint

The NOT NULL Constraint

- Ensures that null values are not permitted for the column
- The NOT NULL constraint can be specified only at the column level, not at the table level.

Syntax :

```
Column_name data_type [CONSTRAINT constraint_name] not null,
```

The UNIQUE Constraint

- A UNIQUE key integrity constraint requires that every value in a column or set of columns (key) be unique.
- UNIQUE constraints allow the input of nulls unless you also define NOT NULL constraints for the same columns
- Defined at either the table level or the column level
- ✓ Column constraint level - Syntax

```
Column_name data_type [CONSTRAINT constraint_name] unique,
```

- ✓ Table constraint level- Syntax

```
…… ,
[CONSTRAINT constraint_name] unique (Column_name) ,
```

The PRIMARY KEY Constraint

- The PRIMARY KEY constraint is a column or set of columns that uniquely identifies each row in a table.
- **Only one primary key can be created for each table** .
- Defined at either the table level or the column level.
- **A composite primary key must be created by using the table-level definition.**

✓ Column constraint level - Example

```
Column_name data_type [CONSTRAINT constraint_name] primary key,
```

✓ Table constraint level- Example

```
…… ,
[CONSTRAINT constraint_name] primary key (Column_name(s)) ,
```

The FOREIGN KEY Constraint

- FOREIGN KEY constraints can be defined at the column or table constraint level
- A composite foreign key must be created by using the table-level definition.

✓ Column constraint level - Example

```
….
department_id NUMBER(4) CONSTRAINT emp_deptid_fk
REFERENCES departments(department_id), ….
```

✓Table constraint level- Example

```
…… ,
CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)
REFERENCES departments(department_id), ……..
```

FOREIGN KEY Constraint Keywords

- FOREIGN KEY: Defines the column in the child table at the table constraint level
- REFERENCES: Identifies the table and column in the parent table
- ON DELETE CASCADE: Deletes the dependent rows in the child table when a row in the parent table is deleted.
- ON DELETE SET NULL: Converts dependent foreign key values to null

Check Constraint

- The CHECK constraint defines a condition that each row must satisfy.
- CHECK constraints can be defined at the column level or table level.
- ✓ Column constraint level - Example

```
….
salary NUMBER(8,2) CONSTRAINT emp_salary_min
CHECK (salary > 0),….
```

- ✓ Table constraint level- Example

```
….
CONSTRAINT emp_salary_min
CHECK (salary > 0),….
```

Comparison Operators

Operator	Meaning
>	Greater Than
<	Less Than
<=	Less Than or equal to
>=	Greater Than or equal to
=	Equal to
<>	Not Equal to

Logical Operators

Operator	Meaning
And	Returns TRUE if *both* component conditions are true
OR	Returns TRUE if *either* component condition is true
NOT	Returns TRUE if the following condition is false

Dropping a Table

- All data and structure in the table is deleted.
- You *cannot* roll back the DROP TABLE statement.
- Syntax:
  - **Drop Table** table_name**;**

Data Dictionary

- See the names of tables owned by the user.
  - SELECT table_name FROM user_tables ;
  - SELECT * FROM Tab;
- View tables, views, synonyms, and sequences owned by the user.
  - Select * from cat;

Renaming a Table

- You rename a table using the RENAME statement

```
RENAME old_table_name TO new_table_name;
```

Alter Table

- You can use ALTER TABLE to perform tasks such as:
  - ✓ Add, modify, or drop a column
  - ✓ Add or drop a constraint.

Alter table – Add Column

Alter Table *TABLE_NAME* **Add** *column_name column_datatype* **[***column1_constraint***];**

```
Alter Table TABLE_NAME Add   (
  column1_name column1_datatype [column1_constraint],
  column2_name column2_datatype [column2_constraint],
  column3_name column3_datatype [column3_constraint]
  );
```

Alter Table – Drop Column

DROP Column :

Alter Table TABLE_NAME **DROP COLUMN** COLUMN_NAME**;**

Alter Table

Modify Column

- **You can change a column's data type, size, and default value.**
- **Guidelines:**
  - You can increase the width or precision of a numeric column.
  - You can increase the width of numeric or character columns.
  - You can decrease the width of a column only if the column contains only null values or if the table has no rows.
  - You can change the data type only if the column contains null values.
  - You can convert a CHAR column to the VARCHAR2 data type or convert a VARCHAR2 column to the CHAR data type only if the column contains null values or if you do not change the size.
  - A change to the default value of a column affects only subsequent insertions to the table.

Alter Table – Modify Column

Modify Column

Alter Table TABLE_NAME **MODIFY** COLUMN_Def**;**

Alter Table – Rename Column

- Rename column of a table

```
ALTER TABLE table_name  RENAME COLUMN old_name to new_name;
```

Adding a Constraint Syntax

Use the ALTER TABLE statement to:

- Add or drop a constraint, but not modify its structure Enable or disable constraints
- **Add a NOT NULL constraint by using the MODIFY clause**

ALTER TABLE *table_name*
ADD [CONSTRAINT *constraint*] *constraint_type* **(***columns***);**

ALTER TABLE *table_name*
DROP CONSTRAINT *constraint_name*;