4. Oracle SQL DDL additional notes(Important fundamentals that will crop up again and again!)

Data Definition language

Naming database objects

Data names in Oracle must conform to the following rules:

- the name must begin with a letter
- maximum length is 30 characters
- names may contain letters, numbers, $, # and _
- can not be an SQL reserved word
- names are not case sensitive

Data names that do not follow the above rules must be enclosed in double quotes ("). For example, "Personnel Listing" would be a valid table name.

Data Types

The basic data types available in Oracle are:

NUMBER(m,n): number containing a total number of m digits with n digits after the decimal.

CHAR(n): a character string of length n. n is a positive integer not exceeding 255. Default is 1.

VARCHAR2(n): a varying-length character string of maximum length n. Maximum length is 4000 bytes.

DATE: a variable containing the date and time; Oracle stores the century, year, month, day, hour, minute and second.

CLOB: character data up to 4 gigabytes

BLOB: binary data up to 4 gigabytes

ROWID: a 64 base number system unique address of table row

TIMESTAMP: date with fractional seconds

Specifying Constraints

Integrity Constraints are used to prevent the entry of invalid information into tables. There are five Integrity Constraints Available in Oracle. They are :

- Not Null
- Primary Key
- Foreign Key
- Check
- Unique

It is essential to follow a naming convention for naming constraints. Oracle recommends the following:

Constraint type	**Abbreviation**
references (foreign key)	fk
unique	uk
primary key	pk
check	ck or (provide detail)
not null	Nn

Format of constraint name

<table name>_<column_name>_<constraint abbreviation>

Table name can be abbreviated.

Constraints can be specified either for a column or for a table. Column constraints are coded after the column entry; for example:

contact_address VARCHAR2(80) NOT NULL

Table constraints are coded after all the column entries, separated by commas.

Constraints can be coded using ALTER TABLE. A combination will be used in these notes and in class.

NOT NULL

By default all columns in a table can contain null values. If you want to ensure that a column must always have a value, i.e. it should not be left blank, then define a NOT NULL constraint on it.

Always be careful in defining NOT NULL constraint on columns, for example if the Personnel table had a marital status attribute some employees might have a partner and some employees might not. If you put NOT NULL constraint on MARITAL_STATUS column then you will not be able insert rows for those employees whose MARITAL_STATUS is null or unknown. Only put NOT NULL constraint on those column which are essential.

PRIMARY KEY

Each table can have one primary key, which uniquely identifies each row in a table and ensures that no duplicate rows exist. Use the following guidelines when selecting a primary key:

- Whenever practical, use a column containing a sequence number. It is a simple way to satisfy all the other guidelines. I will show you how to do this in the practical class
- Database developers tend to minimize use of composite primary keys. Although composite primary keys are allowed and are considered essential (natural PK) they sometimes do not satisfy all of the other recommendations. For example, composite primary key values are long and cannot be assigned by sequence numbers. They must also be specified as a table constraint; for single column keys, either a column or table constraint can be used.
- Choose a column whose data values are unique, because the purpose of a primary key is to uniquely identify each row of the table.
- Choose a column whose data values are never changed. A primary key value is only used to identify a row in the table, and its data should never be used for any other purpose. Therefore, primary key values should rarely or never be changed.
- Choose a column that does not contain any nulls. A PRIMARY KEY constraint, by definition, does not allow any row to contain a null in any column that is part of the primary key.
- Choose an evolvable and security compliant PK.
- It may be necessary to use a surrogate PK.

For example in the Company schema, Personnel table SNUM column is a good candidate for PRIMARY KEY.

Whenever you define a PRIMARY KEY oracle automatically creates a index on that column.

FOREIGN KEY

On whichever column you put FOREIGN KEY constraint then the values in that column must refer to existing values in the other table. A foreign key column can refer to primary key or unique key column of other tables. This Primary key and foreign key relationship is also known as PARENT-CHILD relationship i.e. the table which has Primary Key is known as PARENT table and the table which has Foreign key is known as CHILD table. This relationship is also known as REFERENTIAL INTEGRITY.

Format:

```
   CONSTRAINT constraint-name FOREIGN KEY (col1)
   REFERENCES table-name(primary key) [ON DELETE CASCADE]
```

This clause specifies a column in the child table that must have a corresponding value in the parent table. The foreign key is related to the primary key of the parent table; a primary key must be specified for that table.

REFERENCES table-name: specifies the name of the parent table on which the constraint is being constructed.

ON DELETE CASCADE: specifies that if a row in a parent table is deleted, related rows in the child table are automatically deleted as well.

Oracle allows two other definitions under the definition of the foreign key

The ON DELETE SET NULL action allows data that references the parent key to be deleted, but not updated. When referenced data in the parent key is deleted, all rows in the child table that depend on those parent key values have their foreign keys set to null.

ON DELETE NO ACTION (which is the default) prevents deleting a parent when there are children

CHECK

Use the check constraint to validate values entered into a column.

CONSTRAINT constraint-name CHECK (check-condition) check-condition: a condition involving a column in the table.

Add a check constraint

You can define as many check constraints on a single column as you want there is no restrictions on number of check constraints and that is why it is recommended to give detail on the check constraint, i.e. use MIN, MAX instead of CK.

UNIQUE KEY

Unique Key constraint is same as primary key i.e. it does not accept duplicate values, except the following differences

- There can be only one Primary key per table. Whereas, you can have as many Unique Keys per table as you want.
- Primary key does not accept NULL values whereas, unique key columns can be left blank.

You can also refer to Unique key from Foreign key of other tables.

On which columns should you put Unique Key Constraint ?

It depends on situations, first situation is suppose you have already defined a Primary key constraint on one column and now you have another column which also should not contain any duplicate values, Since a table can have only one primary key, you can define Unique Key constraint on these columns. Second situation is when a column should not contain any duplicate value but it should also be left blank.
For example in the PERSONNEL table SOCIAL_SCEURITY_ID would b a good candidate for Unique Key because all social security numbers are unique but some employees might not have their PPS number available, so you might want to leave this column blank.

To define a UNIQUE KEY constraint on an existing table give the following command.

```
ALTER TABLE PERSONNEL

        ADD CONSTRAINT per_info_social_sec_id_uk

        UNIQUE (social_security_id);
```

Again the above command will execute successfully if social security id column contains complying values otherwise you have to remove non complying values and then add the constraint.

Enabling and Disabling Constraints

You can enable and disable constraints at any time.

To enable and disable constraints the syntax is

```
ALTER TABLE <TABLE_NAME> ENABLE/DISABLE
        CONSTRAINT  <CONSTRAINT_NAME>
```

Dropping constraints.

You can drop constraint by using ALTER TABLE DROP constraint statement.
For example to drop Unique constraint from PERSONNEL table, give the following statement

```
ALTER TABLE PERSONEL

DROP CONSTRAINT per_info_social_sec_id_uk;
```

Altering a Table

After a table has been defined with CREATE TABLE, changes can be made to the structure of the table using the ALTER TABLE command as seen above.

The user can add a column to the table, delete a column or modify the data type of a column; constraints can be added or dropped.

To add a column:

```
ALTER TABLE tablename ADD column-definition
```

The new column is placed logically at the right of the table. Values for the column are initialized to NULL.

To delete a column:

```
ALTER TABLE tablename DROP COLUMN column-name
or
ALTER TABLE tablename DROP (column-name)
```

To modify the description of an existing column, use the following:

```
ALTER TABLE tablename MODIFY column-definition
```

Deleting a Table

To delete a table from the database, the DROP TABLE command is used. This command deletes the table entry and any dependent entries - for example, views, indexes etc. Both the table description and table data are deleted.

```
DROP TABLE table-name
```

Renaming a Table

The RENAME statement can be used to rename a table (this command can also be used for views and synonyms):

```
RENAME old-name TO new-name
```