DEFINING A DATABASE IN SQL

Because of this, the privilege of creating databases may be reserved for the database administrator, and you may need to ask to have a database created.

- Because most systems allocate storage space to contain base tables, views, constraints, indexes, and other database objects when a database is created, you may not be allowed to create a database.
- The basic syntax for creating a database is:

```
CREATE SCHEMA schema_name AUTHORIZATION owner_userid
```

- The database will be owned by the authorized user, although it is possible for other specified users to work with the database or even to transfer ownership of the database.
- Physical storage of the database is dependent on both the hardware and the software environment and is usually the concern of the system administrator.

Several SQL DDL CREATE commands are included in SQL:2016

CREATE SCHEMA	Used to define the portion of a database that a particular user owns. Schemas are dependent on a catalog and contain schema objects, including base tables and views, domains, constraints, assertions, character sets, collations, and so forth.
CREATE TABLE	Defines a new table and its columns. The table may be a base table or a derived table. Tables are dependent on a schema. Derived tables are created by executing a query that uses one or more tables or views.
CREATE VIEW	Defines a logical table from one or more tables or views. Views may not be indexed. There are limitations on updating data through a view. Where views can be updated, those changes can be transferred to the underlying base tables originally referenced to create the view.
CREATE INDEX	Creates a separate data structure that the database management system can use to identify the location of rows that satisfy a specific condition.

- These create commands don't set things in stone they can be reversed (deleted) or changed
  - You can reverse by using the DROP command
    - Thus, DROP TABLE tablename will destroy a table, including its definition, contents, and any constraints, views, or indexes associated with it.
    - DROP SCHEMA or DROP VIEW will also destroy the named schema or view.
  - You can change a table by using the ALTER command
    - ALTER TABLE command can be used to change the definition of an existing base table by adding, dropping, or changing a column or by dropping a constraint.
  - These are other create commands

CREATE CHARACTER SET	Allows the user to define a character set for text strings and aids in the globalization of SQL by enabling the use of languages other than English. Each character set contains a set of characters, a way to represent each character internally, a data format used for this representation, and a collation, or way of sorting the character set.
CREATE COLLATION	A named schema object that specifies the order that a character set will assume. Existing collations may be manipulated to create a new collation.
CREATE TRANSLATION	A named set of rules that maps characters from a source character set to a destination character set for translation or conversion purposes.
CREATE ASSERTION	A schema object that establishes a CHECK constraint that is violated if the constraint is false.
CREATE DOMAIN	A schema object that establishes a domain, or set of valid values, for an attribute. Data type will be specified, and a default value, collation, or other constraint may also be specified, if desired.

Creating Tables

- Here is the general syntax for creating a table (DDL language)

```
CREATE TABLE tablename
( {column definition      [table constraint] } . , . .
[ON COMMIT {DELETE | PRESERVE} ROWS] );

where column definition ::=
column_name
        {domain name | datatype [(size)] }
        [column_constraint_clause. . .]
        [default value]
        [collate clause]

and table constraint ::=
        [CONSTRAINT constraint_name]
        Constraint_type [constraint_attributes]
```

  - Note that in Oracle sqldeveloper, there is a GUI for creating a table which will generate this code for us. What we are referring to here is code itself.
- When preparing to create a table, here is a guideline…
  1. Identify the appropriate data type, including length, precision, and scale, if required, for each attribute.
  2. Identify the columns that should not accept null values. Column controls that indicate a column cannot be null are established when a table is created and are enforced for every update of the table when data are entered.
  3. Identify the columns that need to be unique. When a column control of UNIQUE is established for a column, the data in that column must have a different value for each row of data within that table (i.e., no duplicate values). Where a column or set of columns is designated as UNIQUE, that column or set of columns is a candidate key, as discussed in Chapter 4. Although each base table may have multiple candidate keys, only one candidate key may be designated as a PRIMARY KEY. When a column(s) is specified as the PRIMARY KEY, that column(s) is also assumed to be

NOT NULL, even if NOT NULL is not explicitly stated. UNIQUE and PRIMARY KEY are both column constraints. Note that a table with a composite primary key, OrderLine_T, is defined in Figure 5-6. The OrderLine_PK constraint includes both OrderID and ProductID in the primary key constraint, thus creating a composite key. Additional attributes may be included within the parentheses as needed to create the composite key.

4. Identify all primary key–foreign key connections, as presented in Chapter 4. Foreign keys can be established immediately, as a table is created, or later by altering the table. The parent table in such a parent–child relationship should be created first so that the child table will reference an existing parent table when it is created. The column constraint REFERENCES can be used to enforce referential integrity (e.g., the Order_FK constraint on the Order_T table).
5. Determine values to be inserted in any columns for which a default value is desired. DEFAULT can be used to define a value that is automatically inserted when no value is identified during data entry. In Figure 5-6, the command that creates the Order_T table has defined a default value of SYSDATE (Oracle's name for the current date) for the OrderDate attribute.
6. Identify any columns for which domain specifications may be stated that are more constrained than those established by data type. Using CHECK as a column constraint, it may be possible to establish validation rules for values to be inserted into the database. In Figure 5-6, creation of the Product_T table includes a check constraint, which lists the possible values for ProductFinish. Thus, even though an entry of 'White Maple' would meet the VARCHAR2 data type constraints, it would be rejected because 'White Maple' is not in the checklist.
7. Create the table and any desired indexes, using the CREATE TABLE and CREATE INDEX statements. (CREATE INDEX is not a part of the SQL:2016 standard because indexing is used to address performance issues, but it is available in most RDBMSs.). Indexing is discussed at a more detailed level in Chapter 8 on physical database design.

- Make sure that all constraints, primary key, foreign key, etc are all named properly because it will ensure that you can easily identify these things
- Oracle, MySQL, and some other RDBMSs have an interesting "dummy" table that is automatically defined with each database—the Dual table. The Dual table is used to run an SQL command against a system variable.

```
SELECT Sysdate FROM Dual;
```

displays the current date, and

```
SELECT 8 + 4 FROM Dual;
```

displays the result of this arithmetic expression.

Creating Data Integrity Controls

Referential integrity: a value in the matching column on the many side must correspond to a value in the primary key for some row in the table on the one side or be NULL.

- basically if there isnt a primary key for the foreign key value trying to be added then it wont add it
    - This is called the SQL REFERENCES clause

Suppose we have the scenario where we have a 1:M relationship. The 1 side will have a

primary key which will be referenced by the M side using foreign key. What will happen if we have a record in the 1 side whose primary key is referenced in the M side with a foreign key?

- If a CustomerID value is changed (UPDATE), the connection between that customer and orders placed by that customer will be ruined.
  Solutions:
  - ON UPDATE RESTRICT
    - This problem could be handled by asserting that primary key values cannot be changed once they are established.
    - Then any updates that would delete or change a primary key value will be rejected unless no foreign key references that value in any child table
  - ON UPDATE CASCADE
    - Another solution is to pass the change through to the child table(s)
    - Then, if a customer ID number is changed, that change will flow through (cascade) to the child table, Order_T, and the customer's ID will also be updated in the Order_T table.
  - ON UPDATE SET NULL
    - A third solution is to allow the update on Customer_T but to change the involved CustomerID value in the Order_T table to NULL
    - In this case, using the SET NULL option would result in losing the connection between the order and the customer, which is not a desired effect.
  - What option makes sense?
    - The most flexible option to use would be the CASCADE option.
- What if a customer record was deleted and in turn a CustomerID was deleted?
  Solutions:
  - ON DELETE RESTRICT
    - the customer record could not be deleted unless there were no orders from that customer in the Order_T table.
  - ON DELETE CASCADE
    - removing the customer would remove all associated order records from Order_T.
  - ON DELETE SET NULL
    - the order records for that customer would be set to null before the customer's record was deleted.
  - ON DELETE SET DEFAULT
    - the order records for that customer would be set to a default value before the customer's record was deleted.
  - What option makes sense?
    - ON DELETE RESTRICT would probably make the most sense.

Changing Table Definitions Using ALTER

ALTER TABLE is a way to change or modify a table.

- You can do things like changing base table definitions, add new columns, existing columns can be altered, table contraints can be added or dropped.
    - If columns are to be altered you can include keywords such as ADD, DROP, or ALTER and allow the column's name, data type, length, and constraints to be changed.
    - When the new column is created, it is added to all of the instances in the table, and a value of NULL would be the most reasonable.

Syntax:

-
```
ALTER TABLE table_name alter_table_action;
```

Some of the alter_table_actions available are:

-
```
ADD [COLUMN] column_definition
ALTER [COLUMN] column_name SET DEFAULT default-value
ALTER [COLUMN] column_name DROP DEFAULT
DROP [COLUMN] column_name [RESTRICT] [CASCADE]
ADD table_constraint
```

- Example

Command: To add a customer type column named CustomerType to the Customer table.

```
ALTER TABLE CUSTOMER_T
ADD CustomerType VARCHAR2 (10) DEFAULT 'Commercial';
```

- The ALTER command is invaluable for adapting a database to inevitable modi fications due to changing requirements, prototyping, evolutionary development, and mistakes.

Removing Tables

To remove a table from a database, the owner of the table may use the DROP TABLE command.

Command: To drop a table from a database schema.

-
```
DROP TABLE Customer_T;
```

    - This command will drop the table and save any pending changes to the data base.
    - To drop a table, you must either own the table or have been granted the DROP ANY TABLE system privilege.
    - Dropping a table will also cause associated indexes and privileges granted to be dropped.
- The DROP TABLE command can be qualified by the key words RESTRICT or CASCADE.
    - If RESTRICT is specified, the command will fail, and the table will not be dropped if there are any dependent objects, such as views or constraints, that currently

reference the table.
    - If CASCADE is specified, all dependent objects will (views, constraints, etc) also be dropped as the table is dropped.
- Many RDBMSs allow users to retain the table's structure but remove all of the data that have been entered in the table with its TRUNCATE TABLE command.