Creating tables in SQL

Data manipulation Language – DML

Data definition Language – DDL

Data control Language

Transaction Control Language

**DML**

DML is short name of Data Manipulation Language which deals with data manipulation, and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE etc, and it is used to store, modify, retrieve, delete and update data in database.

* SELECT – retrieve data from the a database
* INSERT – insert data into a table
* UPDATE – updates existing data within a table
* DELETE – Delete all records from a database table
* MERGE – UPSERT operation (insert or update)
* CALL – call a PL/SQL or Java subprogram
* EXPLAIN PLAN – interpretation of the data access path
* LOCK TABLE – concurrency Control

**DDL**

DDL is short name of Data Definition Language, which deals with database schemas and descriptions, of how the data should reside in the database.

* CREATE – to create database and its objects like (table, index, views, store procedure, function and triggers)
* ALTER – alters the structure of the existing database
* DROP – delete objects from the database
* TRUNCATE – remove all records from a table, including all spaces allocated for the records are removed
* COMMENT – add comments to the data dictionary
* RENAME – rename an object

-- DROP DATABASE name\_of\_database --

* CREATE DATABASE bari\_allali\_db
* USE bari\_allali\_db
* CREATE TABLE film\_table
* (
* film\_name VARCHAR(10),
* film\_type VARCHAR(6)
* )
* SELECT \* FROM film\_table;

Datatypes

|  |  |
| --- | --- |
| CHAR(size) | A FIXED length string (can contain letters, numbers, and special characters). The *size* parameter specifies the column length in characters - can be from 0 to 255. Default is 1   * 50% faster than varchar * Use CHAR – fixed length attributes – for example car license plates |
| VARCHAR(size) | A VARIABLE length string (can contain letters, numbers, and special characters). The *size* parameter specifies the maximum column length in characters - can be from 0 to 65535   * ***Memory efficient*** |

|  |  |
| --- | --- |
| INT(*size*) | A medium integer. Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295. The *size* parameter specifies the maximum display width (which is 255) |
| INTEGER(*size*) | Equal to INT(size) |

|  |  |
| --- | --- |
| DATE | A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31' |
| DATETIME(*fsp*) | A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. Adding DEFAULT and ON UPDATE in the column definition to get automatic initialization and updating to the current date and time |

|  |  |
| --- | --- |
| DECIMAL(*size*, *d*) | An exact fixed-point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter. The maximum number for *size* is 65. The maximum number for *d* is 30. The default value for *size* is 10. The default value for *d* is 0. |
| DEC(*size*, *d*) | Equal to DECIMAL(size,d)  Precision and scale |

Float – very large numbers – 8 bytes

|  |  |
| --- | --- |
| BIT(*size*) | A bit-value type. The number of bits per value is specified in *size*. The *size* parameter can hold a value from 1 to 64. The default value for *size* is 1. |
| TINYINT(*size*) | A very small integer. Signed range is from -128 to 127. Unsigned range is from 0 to 255. The *size* parameter specifies the maximum display width (which is 255) |
| BOOL | Zero is considered as false, nonzero values are considered as true. |
| BOOLEAN | Equal to BOOL |

/\*not null modification

ALTER TABLE film\_table

ALTER COLUMN film\_name VARCHAR(10) NOT NULL

Film\_id INT IDENTITY(1,1) PRIMARY KEY – start with 1 increment with 1

Make film\_id a foreign key

FOREIGN KEY (film\_id) REFERENCE film\_table(film\_id)

![A screenshot of a social media post

Description automatically generated]()

A close up of a piece of paper

Description automatically generatedUPDATE command in practice

DELETE command in practice

How to delete a row that has the same value as the foreign key \* homework

Database considerations

* Data security
* Data recovery
* Data integrity
* Normal form

NORMAL FORM



Table\_product\_price

Table\_product\_colour – composite primary key

**First normal form**

make everything atomic and no repeated values

Through normalisation we obtain better performance and efficiency as we are removing redundant values; such as repeated values

**Second normal form**

if first normal form is met then second normal is activated

* All non-key attributes are fully functional dependent on the primary key
* Needs to be fully depended on the composite primary key
* Create another table with the attribute and match it with the missing part of the composite primary key

**Third normal form – requires first and second normal form**

* We do not want a transitive functional dependency
* When a non-key attribute is dependent on another non-key attribute
* Split the table so that the primary key splits the non-key attributes

A screenshot of a social media post

Description automatically generated