SQL

* SQL IS STRUCTED QUERY (COMMAND) LANGUAGE AND WORKS WITH DATABASES.
* SQL IS STANDARD LANGUAGE FOR STORING , MANIPULATING AND RETRIEVING DATA IN DATABASES.
* HOW TO USE SQL IN : MYSQL , SQLSERVER ,MS ACCESS , ORCALE , SYBASE , INFORMIX , POSTGRES , AND OTHER DATABASE SYSTEMS.
* SQL COMMANDS ARE CLASSIFIED INTO DIFFERENT TYPES.

|  |  |  |  |
| --- | --- | --- | --- |
| DATA DEFINITION LANGUAGE (DDL) | DATA MANIPULATION LANGUAGE (DML) | DATA CONTROL LANGUAGE (DCL) | TRANSCTION CONTROL LANGUAGE (TCL) |
| DDL DESCRIBES THE PORTION OF SQL THAT CREATES , ALTERS, AND DELETE OBJECTS. | DML IS A SUBJECT OF OPERATIONS USED TO INSERT , DELETE AND UPDATE DATA IN A DATABSE. | DCL INCLUDES SUCH AS GRANT AND REVOKE. | TCL ARE USED FOR COMMIT AND ROLLBACK. |

DATATYPES IN SQL :-

1.BINARAY 2. NUMERIC 3.STRING 4.DATE AND TIME

(IMAGES) 2.1 APPORIMATE 3.1. CHAR 4.1.DATE

2.1.1.REAL 3.2. VARCHAR 4.2.TIME

2.1.2.FLOAT 3.3.TEXT 4.3.DATE AND TIME

2.2. EXACT

2.2.1.BIT

2.2.2.TINYINT

2.2.3. SMALLINT

2.2.4.INT

2.2.5.BIGINT

2.2.6.DECIMAL

CREATE DATABASE :- TO CREATE DATABASE STATEMENT IS USED TO CREATE A NEW SQL DATABASE.

SYNTAX:-

CREATE DATABASE database\_name ;

USE DATABASE :- TO USE STATEMENT IS USED TO RUN THE DATABASE OF SQL.

SYNTAX :-

USE database\_name:

DROP DATABASE :- TO DROP DATABASE STATEMENT IS USED TO DROP AN EXISTING SQL DATABASE.

SYNTAX :-

DROP DATABASE database\_name ;

SHOW DATABASES :- TO SHOW DATABASE STATEMENT IS USED TO SHOW THE LIST OF DATABASE IN SQL .

SYNTAX :-

SHOW DATABASES;

CREATING TABLES :- TO CREATE TABLE STATEMENT IS USED TO CREATE A NEW TABLE IN A DATABASE.

SYNTAX :-

CREATE TABLE table\_name (  
attribute1 datatype ,

attribute2 datatype,

attribute3 datatype,

……………………………………..);

DROP TABLE :-TO DROP TABLE STATEMENT IS USED TO DROP AN EXISTING TABLE IN A DATABASE.

SYNTAX :-

DROP TABLE table\_name ;

TRUNCATE COMMAND :- IT WILL DELETE THE DATA FROM TABLE.

SYNTAX :-

TRUNCATE TABLE table\_name;

SHOW TABLES :-TO SHOW TABLE STATEMENT IS USED TO SHOW THE LIST OF TABLES WHICH ARE PRESENT IN THE DATABASES .

SYNTAX :-

SHOW TABLES ;

DESC table\_name :- TO DESC STATEMENT IS USED TO DESCRIBE THE TABLES ATTRIBUTE AND DATATYPES .

SYNTAX :-

DESC table\_name ;

RENAME TABLE :-THE RENAME TABLE STATEMENT IS USED TO RENAME THE FILE OF THE TABLE.

SYNTAX :-

RENAME TABLE table\_name before TO table\_name after ;

INSERT INTO :- THE INSERT INTO STATEMENT IS USED TO INSERT NEW RECORDS IN A TABLE. IT IS POSSIBLE TO WRITE THE INSERT INTO TWO WAYS SUCH AS

1. SPECIFIES BOTH THE COLUMN NAMES AND VALUES TO BE INSERTED.

SYNTAX:-

INSERT INTO table\_name (attribute1, attribute2,attribute3,……) VALUES (value1, value2,value3……);

2.INSERT INTO TABLES VALUES BASED ONLY THE VALUES WITHOUT ATTRIBUTES.

SYNTAX:-

INSERT INTO table\_name VALUES (value1,value2,value3,…..);

CONSTRAINTS :-RULES FOR CREATING TABLE ATTRIBUTES WHILE ACCEPT THE DATA.

|  |  |  |
| --- | --- | --- |
| PRIMARY KEY | NOT NULL | CHECK |
| THE PRIMARY KEY CONSTRAINT UNIQUELY IDENTIFIERS EACH IN A TABLE. | THE NOT NULL CONSTRAINT ENFORCES A COLUMN TO NOT ACCEPT NULL VALUES. | THE CHECK CONSTRAINT IS USED TO LIMIT THE VALUE RANGE THAT CAN BE PLACED IN A COLUMN. |
| PRIMARY KEYS MUST CNTAINT UNIQUE VALUES AND CANNOT CONTAIN NULL VALUES. | THIS ENFORCES A FIELD TO ALWAYS CONTAIN A VALUE , WHICH MEANS THAT YOU CANNOT INSERT A NEW RECORD. | IT WILL CHECK THE CONSTRAINT ON A COLUMN. IT WILL ALLOW ONLY CRETAIN VALUES FOR THIS COLUMN. |
| SYNTAX :-  CREATE TABLE tb\_name(  ATT1 DATATYPE,  ATT2 DATATYPE,  ATT3 DATATYPE,  PRIMARY KEY(ATT1)); | SYNTAX :-  CREATE TABLE tb\_name(  ATT1 DATATYPE NOT NULL,  ATT2 DATATYPE,  ATT3 DATATYPE); | SYNTAX :-  CREATE TABLE tb\_name(  ATT1 DATATYPE,  ATT2 DATATYPE,  ATT3 DATATYPE  CHECK(ATT2>=18)); |

|  |  |  |
| --- | --- | --- |
| DEFAULT | UNIQUE | FOREIGN KEY |
| THE DEFAULT CONSTRAINT IS USED TO SET A DEFAULT VALUE FOR COLUMN. | THE UNIQUE CONSTRAINT ENSURES THAT ALL VALUES IN A COLUMN ARE DIFFERENT. | THE FOREIGN KEY CONSTRAINT IS USED TO RELATE OR LINK BETWEEN TWO TABLES. |
| THE DEFAULT VALUE WILL BE ADDED TO ALL NEW RECORDS , IF NO OTHER VALUE IS SPECIFIED | A PRIMARY KEY CONSTRAINT AUTOMATICALLY HAS A UNIQUE CONSTRAINT.NO DUPLICATE BUT ACCEPTS NULL VALUES | A FOREIGN KEY IS A COLLECTION OF FIELDS IN ONE TABLE THAT REFERS TO THE PRIMARY KEY IN ANOTHER TABLE. |
| SYNTAX:-  CREATE TABLE tb\_name(  ATT1 DATATYPE,  ATT2 DATATYPE DEFAULT ‘VALUE ’ ); | SYNTAX :-  CREATE TABLE tb\_name(  ATT1 DATATYPE,  ATT2 DATATYPE UNIQUE); | SYNTAX :-  CREATE TABLE tb\_name(  ATT1 DATATYPE,  ATT2 DATATYPE,  FOREIGN KEY (ATT1) REFERENCES tb\_name(ATT1)); |

AUTO\_INCREMENT :- AUTO INCREMENT ALLOWS A UNQIUE NUMBER TO BE GENERATED AUTOMATICALLY WHEN A NEW RECORD IS INSERTED INTO TABLE. BY DEFAULT, THE STARTING VALUE FOR AUTO\_INCREMENT IS 1 AND IT WILL INCREMENT BY 1 FOR EACH NEW RECORD WE CAN ALSO SET THE DEFUALT VALUE WITH AUTO\_INCREMENT = VALUES .

SYNTAX:-

CREATE TABLE table\_name(  
ATTRIBUTE1 DATATYPE PRIMARY KEY AUTO\_INCREMENT ,

ATTRIBUTE2 DATATYPE );

JOINS :- A JOIN CLAUSE IS USED TO COMBINE ROWS FROMS TWO OR MORE TABLES , BASED ON A RELATED COLUMN BETWEEN THEM. DIFFERENT TYPES OF SQL JOINS :

|  |  |
| --- | --- |
| (INNER) JOIN | RETURNS RECORDS THAT HAVE MATCHING VALUES IN BOTH TABLES. |
| LEFT (OUTER) JOIN | RETURNS ALL RECORDS FROM THE LEFT TABLE , AND THE MATCHED RECORDS FROM THE RIGHT TABLE. |
| RIGHT (OUTER) JOIN | RETURNS ALL RECORDS FROM THE RIGHT TABLE AND THE MATCHED RECORDS FROM THE LEFT TABLE. |
| FULL (OUTER) JOIN | RETURNS ALL RECORDS WHEN THERE IS A MATCH IN EITHER LEFT OR RIGHT TABLE. |

SYNTAX:-

SELECT \* FROM table\_name1 INNER JOIN table\_name2 ON table\_name1.attribute = table\_name2.attribute;

SELECT col1, col2, col3… FROM table\_name1 INNER JOIN table\_name2 ON table\_name1.attribute = table\_name2.attribute;

FUNCTION IN SQL :- A FUNCTION IS A STORED PROGRAM WE CAN PASS PERMETER TO GET A VALUE.

|  |  |
| --- | --- |
| COUNT() | WHICH WILL GIVE THE COUNT OF VALUES OF AN ATTRIBUTE. |
| SUM() | IT WILL APPLY FOR NUMERICAL VALUES. SUM OF ALL VALUES OF ATTRIBUTE. |
| AVG() | AVERAGE OF ALL VALUES |
| MIN() | MINIMUM VALUES OF AN ATTRIBUTE |
| MAX() | MAXIMUM VALUES OF AN ATTRIBUTE |
| CONCAT() | IT WILL COMBINES TWO TABLES OF THE DATA.  CONCAT(ATTRIBUTE1, ATTRIBUTE2) , CONCAT(ATTRIBUTE1,” “, ATTRIBUTE2) |

SYNTAX :-

SELECT FUNCTION(ATTRIBUTE) FROM table\_name;

SELECT FUNCTION(ATTRIBUTE) AS default name FROM table\_name;

ALTER COMMAND IN SQL :- TO CHANGE THE SCHEMA OR STRUCTURE OF DATABASE.

1. ALTER TABLE -ADD COLUMN :- TO USE ADD COLUMN IN A TABLE .

SYNTAX :-

ALTER TABLE table\_name

ADD column\_name datatype;

1. ALTER TABLE- DROP COLUMN :- TO USE DELETE A COLUMN FROM THE TABLE.

SYNTAX :-

ALTER TABLE table\_name  
DROP COLUMN column\_name;

1. ALTER TABLE-DROP CONSTRAINT :- TO USE DELETE A CONSTRAINT FROM THE TABLE ATTRIBUTE.

SYNTAX :-

ALTER TABLE table\_name

DROP CONSTRAINT\_NAME;

1. ALTER TABLE-MODFIY COLUMN :- TO CHANGE THE DATE TYPE OF COLUMN IN A TABLE ATTRIBUTE.

SYNTAX:-

ALTER TABLE table\_name  
MODIFY COLUMN column\_name DATATYPE;

1. ALTER TABLE-RENAME COLUMN :- TO CHANGE THE NAME OF A COLUMN IN A TABLE.

SYNTAX:-

ALTER TABLE table\_name  
RENAME COLUMN column\_name before TO column\_name after ;

1. ALTER TABLE-MODIFY POSITIONS :- TO CHANGE THE POSITIONS BASED OF THE COLUMNS.

SYNTAX :-

ALTER TABLE table\_name  
MODIFY column\_name DATATYPE AFTER/ LAST/FIRST column\_name;

1. ALTER TABLE-ADD CONSTRAINT :-TO ADD CONSTRAINT TO ATTRIBUTE OF THE TABLES.

SYNTAX :-

ALTER TABLE table\_name

ADD CONSTRAINT constraint(attribute);

1. ALTER TABLE-DROP CONSTRAINT :- TO DROP THE CONSTRAINT OF THE COLUMNS IN THE TABLES.

SYNTAX:-

ALTER TABLE table\_name

DROP constraint\_name;

SYNTAX :- FOR DROP OF UNIQUE CONSTRIANT

ALTER TABLE table\_name

DROP INDEX ATTRIBUTE;

SELECT COMMAND IN SQL :- TO RETRIEVE DATA FROM THE TABLE.

1.SELECT COLUMN1, COLUMN2,……FROM table\_name;

2.SELECT \* FROM table\_name;

3.SELECT DISTINCT column1,column2,…….FROM table\_name;

4.SELECT column1 , column2, ……. FROM table\_name WHERE condition;

5.SELECT column1, column2,…….FROM table\_name WHERE condition AND condition ;

6. .SELECT column1, column2,…….FROM table\_name WHERE condition OR condition ;

7. SELECT column1 , column2, ……. FROM table\_name WHERE NOT condition;

8.SELECT column1,column2…..FROM table\_name ORDER BY column1,column2,….ASC/DESC;

9. SELECT column1 , column2, ……. FROM table\_name WHERE condition LIMIT number;

10.SELECT \* FROM table\_name WHERE condition ORDER BY column DESC LIMIT number;

11.SELECT \* FROM table\_name WHERE condition LIKE pattern;

12.SELECT column\_name FROM table\_name WHERE column\_name IN (value1,value2….);

13.SELECT column\_name FROM table\_name WHERE column\_name BETWEEN value1 AND value2;

SQL OPERATIONS :-

|  |  |
| --- | --- |
| OPERATOR | DESCRIPTION |
| = | EQUAL |
| > | GREATER THAN |
| < | LESS THAN |
| >= | GREATER THAN OR EQUAL |
| <= | LESSER THAN OR EQUAL |
| <> | NOT EQUAL |
| != | NOT EQUAL IS SOME VERSIONS |
| AND | BOTH CONDITION MUST BE TRUE |
| OR | EITHER CONDITION CAN BE TRUE |
| NOT | TRUE 🡪 FALSE AND FALSE 🡪 TRUE |
| ORDER BY | THE ORDER BY KEYWORD IS USED TO SORT THE RESULT-SET IN ASCENDING OR DESCENDING ORDER. BY DEFAULT THE ORDER BY IS ASSCENDING ORDER AND DESC IS THE KEY TO DESCENDING ORDER. |
| LIMIT | WE CAN LIMIT THE DATA FROM TABLE. |
| LIKE | THE LIKE OPERATOR IS USED IN A WHERE TO SEARCH FOR A SPECIFIED PATTERN IN A COLUMN. |
| IN | THE IN OPERATOR ALLOWS YOU TO SPECIFY MULTIPLE VALUES IN A WHERE CLAUSE. |
| BETWEEN | THE BETWEEN OPERATOR SELECTS VALUES WITHIN A GIVEN RANGE. THE VALUES CAN BE NUMBERS, TEXT , AND DATES . |

WHERE CLAUSE :- THE WHERE CLAUSE IS USED TO FILTER RECORDS. IT USED TO EXACT RECORD ONLY THOSE WHO HAS SPECIFIED CONDITIONS.

SYNTAX:-

SELECT column1 , column2, ……. FROM table\_name WHERE condition;

CURRENT\_DATE() ,CURRENT\_TIME() AND NOW() :- HERE THE VALUES OF THE SQL DATE IS TAKEN BASED ON THE CURRENT\_DATE() , TIME IS TAKEN BASED ON THE CURRNET\_TIME() AND NOW () MEANS IT WILL TAKE BOTH CURRENT DATE AND TIME;

SYNTAX :-

INSERT INTO table\_name

VALUE (CURRENT\_DATE() , CURRENT\_TIME() , NOW());

WILD CARDS :- USED TO SUBSTITUTE ONE OR MORE CHARACTER IN A STRING.

|  |  |
| --- | --- |
| A% | FINDS ANY VALUES STARTS WITH “A” |
| %A | FINDS ANY VALUES THAT ENDS WITH “A” |
| %OR% | FINDS ANY VALUES THAT HAVE “OR” IN ANY POSITION |
| \_R% | FINDS ANY VALUES THAT HAVE “R” THE SECOND POSITION. |
| A\_% | FINDS ANY VALUES THAT STARTS WITH “A” AND ARE ATLEAST 3 CHARACTERS. |
| A%O | FINDS ANY VALUES THAT STARTS WITH “A” AND ENDS WITH “O”. |
| \_\_\_\_-\_\_-\_\_ | FINDS ANY VALUES BASED ON THE DATE. |

UNION :- THE UNION OPERATOR IS USED TO COMBINE THE RESULT-SET OF TWO OR MORE SELECT STATEMENTS.EVERY SELECT STATEMENT WITHIN UNION MUST HAVE THE SAME NUMBER OF COLUMNS. THE COLUMNS MUST ALSO HAVE SIMILAR DATA TYPES. THE COLUMNS IN EVERY SELECT STATEMENT MUST ALSO BE IN SAME ORDER. TO ALLOWS DUPLICATE VALUES , USE UNION ALL.

SYNTAX :-

SELECT column\_name(s) FROM table1 UNION SELECT column\_name FROM table2;

SELECT column\_name(s) FROM table1 UNION ALL SELECT column\_name FROM table2;

SELF JOINS :-

* JOINS ANOTHER COPY OF A TABLE TO ITSELF.
* USED TO COMPARE ROWS OF THE SAME TABLE.
* HELPS TO DISPLAY A HEIRARCHY OF DATA.

SYNTAX :-

SELECT \* FROM table1 INNER JOINS tablE2 ON conditions.

VEIWS :-

* A VIRTUAL TABLE BASED ON THE RESULT-SET OF AN SQL STATEMENT.
* THE FIELDS IN A VIEW ARE FIELDS FROM ONE OR MORE REAL TABLES IN THE DATABASE.
* THEY ARE NOT REAL TABLES BUT CAN BE INTERACTED WITH AS IF THEY WERE ANOTHER TABLE.

SYNTAX :-

CREATE VEIW view\_name AS SELECT column1,column2,….. FROM table\_name ;

DROP VIEW :- TO USE DROP VEIW STATEMENT , WE CAN DELETE THE VEIW OF THE COLUMN IN A TABLE.

SYNTAX :-

DROP VEIW view\_name ;

INDEXES :-

* INDEXES ARE USED TO FIND VALUES WITHIN A SPECIFIES COLUMN MORE QUICKLY.
* MYSQL NORMALLY SEARCHES SEQUENTIALLY THROUGH A COLUMN.
* THE LONGER THE COLUMN, THE MORE EXPENSIVE THE OPERATION IS
* UPDATE TAKES MORE TIME ; SELECT TAKES LESS TIME.

SYNTAX :-

SHOW INDEXES FROM table\_name ;

CREATE INDEX indexes\_name ON table\_name(column\_name);

DROP INDEX :- TO DELTE THE INDEXES OF THE TABLE.

ALTER TABLE table\_name

DROP INDEX indexes\_name;

SUBQUERY :- A QUERY WITH ANOTHER QUERY I.E. QUERY(SUBQUERY) .

SYNTAX :-

SELECT COLUMN1, COLUMN2,……FROM table\_name WHERE CONDITION > (SELECT COLUMN1,…FROM table\_name WHERE CONDITION )

ANY & ALL OPERATORS :-

THEY ARE ANY AND ALL OPERATORS ALLOW TO PERFORM A COMPARSION BETWEEN A SINGLE COLUMN VALUE AND A RANGE OF OTHER VALUE.

|  |  |
| --- | --- |
| ANY OPERATOR | ALL |
| ANY MEANS THAT THE CONDITION WILL BE TRUE IF THE OPERATION IS TRUE FOR ANY OF THE VALUES IN THE RANGE. | ALL MEANS THAT THE CONDITION WILL BE TRUE ONLY IF THE OPERATION IS TRU FOR ALL VALUES IN THE RANGE. |
|  |  |
|  |  |

GROUP BY CLAUSE:- AGGREGATE ALL ROWS BY A SPECIFIC COLUMN OFTEN USED WITH AGGREGATE FUNCTION (SUM() , MAX() , MIN() ,AVG() , COUNT() ) ;

SYNTAX:-

SELECT FUCNTION (column\_name) , column\_2 ,…

FROM table\_name

GROUP BY column\_name ;

HAVING CLAUSE :- THE HAVING CLAUSE WAS ADDED TO SQL BECAUSE THE WHERE KEYWORD CANNOT BE USED WITH AGGREGATE FUNCTIONS.

SYNTAX :-

SELECT FUNCTION(column\_name) , column\_name FROM table\_name

GROUP BY column\_name

HAVING CONDITION ;

ROLL UP :- EXTENSION OF THE GROUP BY CLAUSE PRODUCES ANOTHER ROW AND SHOWS THE GRAD TOTAL (SUPER-AGGREGATE VALUE).

SYNTAX :-

SELECT FUCNTION (column\_name) , column\_2 ,…

FROM table\_name

GROUP BY column\_name WITH ROLL UP ;

ON DELETE :-

* + ON DELETE SET NULL = WHEN A FOREIGN KEY IS DELETED , REPLACE FOREIGN KEY WITH NULL.
  + ON DELETE CASCADE = WHEN A FOREIGN KEY , DELETE ROW.

SYNTAX :-

STORED PROCEDURE :- IT IS PREPARED SQL CODE THAT YOU CAN SAVE IF THERE’S A QUERY THAT YOU WRITE OFTEN.

* REDUCES NETWORK TRAFFIC .
* INCREASE PERFORMANCE .
* SECURE , ADMIN CAN GRANT PERMISSION TO USE.
* INCREASE MEMORY USAGE OF EVERY CONNECTION.

SYNTAX :-

DELIMITER $$

CREATE PROCEDURE procedure\_name

BEGIN

SLECT \* FROM table\_name ;

END $$

DELIMITER;

CALL procedure\_name();

DROP PROCEDURE procedure\_name();

DELIMITER $$

CREATE PROCEDURE procedure\_name (IN name DATATYPE , IN name DATATYPE)

BEGIN

SLECT \*

FROM table\_name

WHERE CONDITION ;

END $$

DELIMITER;

CALL procedure\_name(name);

DROP PROCEDURE procedure\_name();

TRIGGER :- WHEN AN EVENT HAPPENS , DO SOMETHING

EX :- (INSERT ,UPDATE, DELETE)

CHECKS DATA , HANDLES ERRORS , AUDITING TABLES

SYNTAX :-