Contents

[ALTER (RENAME, MODIFY) 3](#_Toc170976073)

[Dimensions of ALTER-ADD 3](#_Toc170976074)

[LIKE OPERATOR 4](#_Toc170976075)

[GENERAL PATTERNS 4](#_Toc170976076)

[CREATE TABLE 5](#_Toc170976077)

[Basic syntax for CREATE 5](#_Toc170976078)

[DESC command 6](#_Toc170976079)

[AND-OR 6](#_Toc170976080)

[DISTINCT in DBMS 8](#_Toc170976081)

[IN 8](#_Toc170976082)

[Various Dimensions of IN 8](#_Toc170976083)

[Aggregate Functions and Group By 9](#_Toc170976084)

[Group By 9](#_Toc170976085)

[Differences between where and having clause 9](#_Toc170976086)

[ORDER BY Clause 10](#_Toc170976087)

[WHERE in SQL 11](#_Toc170976088)

[DROP-TRUNCATE-RENAME 12](#_Toc170976089)

[Join in SQL 12](#_Toc170976090)

[Normalization in DBMS 13](#_Toc170976091)

ALTER-ADD

* Used to add one or more new columns to the existing database tables.

ALTER TABLE table\_name ADD(column\_name datatype);

Dimensions of ALTER-ADD

1. Adding a single column
2. Adding multiple columns
3. Adding constraints

ALTER TABLE emp ADD(ADDRESS VARCHAR2(15));

SELECT \* FROM EMP

ALTER TABLE EMP ADD(DOB DATE, FATHERNAME VARCHAR(12), MOTHERNAME VARCHAR(12), ALLOWANCES NUMBER(4));

We are adding 5 columns

ALTER TABLE EMP

ADD PRIMARY KEY(EMPNO);

ADDING A COLUMN WITH DEFAULT VALUE

ALTER TABLE EMP

ADD(gender CHAR(1) DEFAULT “F”)

ALTER-DROP

* Alter command is used to delete one or more existing columns present in the table
* We can also drop more than one column by separating the column names with commas

ALTER TABLE table\_name DROP(column\_name);

Dimensions of Alter-drop

1. Dropping a single
2. Dropping multiple columns
3. Dropping constraints

ALTER TABLE EMP

DROP(DOB);

ALTER TABLE EMP

DROP(MOTHERNAME, FATHERNAME, ALLOWANCES);

ALTER TABLE EMP

DROP CONSTRAINT PRIMARY\_ID

## ALTER (RENAME, MODIFY)

SQL ALTER COMMAND

It is used to add columns, delete columns, drop constraints, renaming the columns, changing the data type and data type size of the column existing in the table

### Dimensions of ALTER-ADD

1. Adding a single column
2. Adding multiple columns
3. Adding constraints

ALTER-DROP

* Alter command is used to delete one or more existing columns present in the table
* We can also drop more than one column by separating the column name with commas

ALTER TABLE table\_name DROP(column\_name);

Dimensions of ALTER-DROP

1. Dropping a single column
2. Dropping multiple columns
3. Dropping constraints

ALTER-MODIFY

* We can change the data type of the existing column or the size of the data type of the existing column

ALTER TABLE table\_name MODIFY (column\_name datatype);

Dimensions of ALTER-MODIFY

1. Modifying a single column
2. Modifying multiple columns

ALTER TABLE EMP MODIFY( ADDRESS VARCHAR2(75));

ALTER-RENAME

* Used to rename the existing column name of a table

ALTER TABLE table\_name RENAME old\_column\_name TO new\_column\_name;

ALTER TABLE EMP RENAME COLUMN ADDRESS TO LOCATION1;

## LIKE OPERATOR

LIKE operator used to search specified pattern in the data and retrieve the record when there is a pattern match as required

### GENERAL PATTERNS

PATTERNS MEANINGS

‘r%’ Matches strings which start with ‘r’

‘%r’ Matches strings with end with ‘r’

‘r%t’ Matches strings which contain the start with ‘r’ and

end with ‘t’

‘%tri%’ Matches strings which contain the substring ‘tri’

‘\_tri%’ Matches strings which contain the substring ‘ tri’ in

them at the second position

‘\_r%’ Matches strings which contain ‘r at the second

position

‘r\_\_%’ Matches strings which start with ‘r’ and contain at

least 2 more characters.

Example Queries

* Display the employees whose

1. Name start with “M”
2. Ends with “M”

* Display the names of all employees having M in any position in their name
* Display the name of all employees which contains the fourth letter as M

SELECT ename FROM emp

Where ename LIKE ‘M%’;

SELECT ename FROM emp

WHERE ename like ‘%M’;

Select ename from emp where ename like ‘%m%’

* Display the employee names and hire dates for the employees joined in the month of December
* Display names of all employees whose name contains exactly 4 letters
* Display the names of all employees whose name does not contain ‘A’ anywhere

SELECT ENAME, HIREDATE FROM EMP WHERE HIREDATE LIKE ‘%DEC%’

SELECT ENAME FROM EMP WHERE ENAME LIKE ‘\_\_\_\_’;

SELECT ENAME FROM EMP WHERE ENAME LIKE ‘\_L%’

SELECT ENAME FROM EMP WHERE ENAME NOT LIKE ‘%L%’;

## CREATE TABLE

* Create command is used to create a table in the database with the structure specified by the user
* This structure includes the number of columns to be present in the table and the data type of the column, size of data, etc

### Basic syntax for CREATE

CREATE TABLE table\_name

(

column1 datatype(size),

column2 datatype(size),

column3 datatype(size),

…

columnN datatype(size),

PRIMARY KEY(one or more columns)

);

Create table employee(

empno number(4,0),

ename varchar2(10),

job varchar2(9),

### DESC command

Structure of the table that we have created

* Column names,
* Data types of column names,
* Size of data,
* Any constraints imposed the table,
* Default values for each column,
* Whether null values allowed or not for each column, etc

Creating a new table from an existing table

* By copying all columns from another table
* By copying selected columns from another table
* By copying selected rows from another table
* Copying selected column from multiple tables

Select \* from emp

Select \* from employee table

Create table emp1 as (select \* from emp)

Select \* from emp1

Create table emp2 (select empno, ename, sal from emp)

Create table emp3 as (select \* from emp where deptno = 10);

Create table emp4 as

(select ename, job, sal, loc dname from emp, dept);

### AND-OR

AND/OR in DBMS

The AND and OR operators are used with the where clause for precise filtration of data from the database tables by combining more than one condition along with select, update and delete queries.

AND clause

* Definition: The AND results true only when all the conjunction of conditions specifies after the where clause are satisfied

SELECT column1, column2, …

FROM table\_name

WHERE condition1 AND condition2 AND condition3…;

Practice

* Display the records of those employees who are working as a manager and getting salary greater that 2500/-

Select job, ename, sal from emp

Where job = ‘MANAGER’ AND SAL>2500

OR clause

* Definition: Among multiple conditions specified in the where clause the transaction is performed if any of the condition becomes true

SELECT column1, column2, …

FROM table\_name

WHERE condition1 OR condition2 OR condition3 …

Practice

* Display details of all analyst and managers

Select job, ename, sal

From emp

WHERE JOB= ‘MANAGER’ JOB= ‘ANALYST’

Combining AND and OR

* Displays the details of employees having a salary greater than 1500/-
* Or if he is a manager which is exclusively from 10th dept and 30th departments.

SELECT job, ename, sal from emp

Where (sal >1500 or job = ‘MANAGER’) AND (DEPT = 10 OR DEPT = 30)

## DISTINCT in DBMS

DISTINCT statement is used to return only unique values present in a column or combination of columns

SELECT DISTINCT column1, column2, FROM table\_name;

Dimensions of DISTINCT

1. DISTINCT on a single column
2. DISTINCT on more than one column

Select \* from emp;

Select job from emp;

Select distinct job from emp;

Select job, sal from emp;

Select distinct job, sal from emp; // this returns any field with distinct values

Distinct rows are returned

Select job, count (\*) from emp

Group by job

Having count (\*)>1

Does the DISTINCT clause consider NULL to be a unique value in SQL?

YES

## IN

* The SQL IN condition (sometimes called the IN operator) allows you to easily test if an expression matches any value in a list of values.
* It is used to help reduce the need for multiple OR conditions in a SELECT, INSERT, UPDATE, or DELETE statement.

### Various Dimensions of IN

* Using the IN condition with character values
* Using the IN condition with Numeric values
* Using the IN condition with the NOT operator
* Performing DML operations using IN using subquery as an expression inside IN

select \* from emp where ename= ‘BLAKE’ OR ENAME= ‘SCOTT’ OR ENAME= ‘FORD’;

SELECT \* FROM EMP WHERE ENAME IN(‘BLAKE’, ‘SCOTT’, ‘FORD’);

SELECT \* FROM EMP WHERE EMPNO IN(7369, 7844,7876);

SELECT \* FROM EMP WHERE EMPNO NOT IN (7369, 7844,7876);

DELETE FROM EMP

WHERE ENAME IN(‘JAMES’, ‘ADAMS’, ‘MILLER’)

SELECT MAX(SAL) FROM EMP

WHERE SAL NOT IN(SELECT MAX(SAL) FROM EMP);

## Aggregate Functions and Group By

Aggregate Functions

SUM

MAX

MIN

COUNT

AVG

SELECT MAX(marks) FROM student;

Output: 100

NULL IS IGNORED

### Group By

We use group by whenever we want to group rows according to one or more attributes

1. Find maximum marks in every subject

SELECT subject, MAX (marks) FROM student GROUP BY subject

1. Find sum of marks of the students having sum more than 150

SELECT name, SUM(marks) AS msum

FROM student

GROUP BY name

HAVING SUM(marks) >150

### Differences between where and having clause

## ORDER BY Clause

The order by clause is used to arrange the fetched data from the database table in ascending or descending order of data values based on one or more columns

For Example,

* Store records of employees in increasing order of salaries
* Student details as per increasing order of marks or attendance
* Employee details based on seniority level

Sorting according to a single column

* Display the names, salaries of all employees based on decreasing order of their salaries
* Display the department number and employee name as per increasing order of department numbers
* Display the all the details of employees as per increasing order of (i) department numbers (ii) hire date

SELECT ename, sal

FROM emp

ORDER BY sal DESC

SELECT ename, sal, deptno

FROM emp

ORDER BY deptno;

SELECT \* FROM emp

ORDER BY ename;

Sorting according to more than one column

* Display the details of employees based on increasing order of Departments and in each department, salary should be further arranged in highest to lowest order.

SELECT \* from emp

Order by deptno, sal;

Using where clause in ORDER BY

* Display the names, salary, jobs of employees who are working as manager in highest to lowest order.

Select ename, sal, job from emp

Where job = ‘MANAGER’

order by sal desc;

Specifying column numbers in ORDER BY

* Instead of column names, you can also use the position number of columns are specified in the select statement.

Select \* from emp order by 8, 6;

// here 8 is the department number and 6 is the column of salary

Sorting data based on expressions

* Display the names, job, annual salary and hire date of all employees based on decreasing order of their annual salary.

Select ename, job, sal, hiredate, sal\* 12 as ann\_sal from emp

Order by sal desc;

## WHERE in SQL

Sometimes a user is interested in filtering data and retrieving only specific data that meet certain conditions and requirements, which can be done by where clause which performs data filtering

SELECT column1, column2, FROM table\_name WHERE [condition];

Simple examples,

1. Display the records of those employees whose designation is ‘MANAGER’
2. Display name, job, the salary of that employee whose salary is greater than 2900

SELECT \* FROM emp

WHERE job= ‘MANAGER’

SELECT ename, sal, job FROM emp

WHERE sal > 2900

Operators that can be used in WHERE clause Aggregate Functions and Group By

1. Display the records of all employees other than managers
2. Display records of all employees whose salary is between 1000/- to 2500/-

SELECT enamem, sal, job FROM emp WHERE job != ‘MANAGER’

SELECT ename, sal, job FROM emp WHERE sal between 1000 and 2500;

// for between operator place the lowest value first then the higher value.

Deleting records based on the condition specified in where clause.

DELETE FROM emp

WHERE sal = 2450;

## DROP-TRUNCATE-RENAME

DROP COMMAND

* Definition: Drop command delete the table existence completely. i.e. drop statement destroys the existing database object of that particular table, index or view

DROP TABLE table\_name;

Restoring a dropped table using FLASHBACK command

FLASHBACK TABLE table\_name TO BEFORE DROP

Dropping a table permanently: PURGE command

DROP TABLE table\_name PURGE;

TRUNCATE command

* The truncate command will make the table empty i.e. all the table data will be deleted but the structure and database object is still alive and the table can be reused normally

Syntax

TRUNCATE TABLE table\_name;

TRUNCATE TABLE emp;

Renaming a table: RENAME command

Syntax

RENAME old\_table\_name to new\_table\_name;

## Join in SQL

Normalization means we split the data into multiple tables.

Employee

|  |  |
| --- | --- |
| Name | Dept\_ID |
| ABC | 3 |
| Xyz | 1 |
| PQR | 1 |
| BCD | NULL |

Department

|  |  |
| --- | --- |
| Dept\_ID | Dept\_name |
| 1 | Engineering |
| 2 | Sales |
| 2 | Marketing |

Different types of Join

* Cross product is going to have 12 rows
* Inner join – we use a constraint. Only those rows in the cross product that satisfies that constraint are included in the result
* Eque join is a type of inner join where equal operator is used to join two tables.
* Natural Join is a special case of an eque join where you are doing join on the columns that have the same name.
* Outer join
* Left outer
* Right outer
* Full
* Self join is a special case of join where you want to join the table to it self.

## Normalization in DBMS

Normalization is a technique of organizing the data into multiple related tables, to minimize DATA REDUNDANCY.

What is data redundancy? And why should we reduce it?

Data redundancy is the repetition of data in multiple places.

Drawback of data redundancy:

* Repetition of data increases the size of data base.
* Other issues like:
  + Insertion anomaly
  + Deletion anomaly
  + Updating anomaly

Before Normalization

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Rollno | Name | Branch | Hod | Office\_tel |
| 1 | Akon | Cse | Mr.x | 53337 |
| 2 | Bkon | cse | Mr.x | 53337 |
| 3 | Ckon | Cse | Mr.x | 53337 |
| 4 | Dkon | Cse | Mr.x | 53337 |

After Normalization

|  |  |  |
| --- | --- | --- |
| Rollno | Name | Branch |
| 1 | Akon | Cse |
| 2 | Bkon | cse |
| 3 | Ckon | Cse |
| 4 | Dkon | Cse |

|  |  |  |
| --- | --- | --- |
| Branch | Hod | Office\_tel |
| Cse | Mr.x | 53337 |

Issues

Insertion Anomaly:

To insert redundant data for every new row (of Student in our case) is a data insertion problem or anomaly.

Deletion Anomaly:

Loss of a related dataset when some other dataset is deleted.

Modification Anomaly:

Mr. X leaves, and Mr. Y joins as the new HOD for CSE

How normalization will solve the problem?

Breaks the student table into two tables

Student table + branch table