06-06-2025

Contents

[WHY DATABASE? 3](#_Toc200622249)

[WHAT IS A DATABASE? 3](#_Toc200622250)

[CHARECTERISTICS OF A GOOD DATABASE 3](#_Toc200622251)

[DATA MODELS 4](#_Toc200622252)

[OBJECT BASED LOGICAL MODEL 4](#_Toc200622253)

[RECORD BASED LOGICAL MODEL 4](#_Toc200622254)

[PHYSICAL DATA MODEL 5](#_Toc200622255)

[RELATIONAL MODEL TERMINOLOGY 5](#_Toc200622256)

[DDL-DATA DEFENITION LANGUAGE 5](#_Toc200622257)

[WHY DDL? 5](#_Toc200622258)

[DATA TYPE 5](#_Toc200622259)

[ALTER 6](#_Toc200622260)

[ ALTER-MODIFY 6](#_Toc200622261)

[ ALTER ADD 6](#_Toc200622262)

[ ALTER RENAME 6](#_Toc200622263)

[ ALTER DROP 6](#_Toc200622264)

[Problem 6](#_Toc200622265)

[RENAME 7](#_Toc200622266)

[TRUNCATE: 7](#_Toc200622267)

[CONSTRAINTS 7](#_Toc200622268)

[What is a constraint 7](#_Toc200622269)

[Not Null 8](#_Toc200622270)

[Unique 8](#_Toc200622271)

[Check 8](#_Toc200622272)

[primary key 8](#_Toc200622273)

[foreign key 9](#_Toc200622274)

[on delete cascade 9](#_Toc200622275)

[on delete set null 10](#_Toc200622276)

[DML-DATA MANIPULATION LANGUAGE 10](#_Toc200622277)

[INSERT 10](#_Toc200622278)

[UPDATE 10](#_Toc200622279)

[DELETE 10](#_Toc200622280)

[Functions 10](#_Toc200622281)

[Single row functions 10](#_Toc200622282)

[Numeric Functions 10](#_Toc200622283)

[STRING FUNCTIONS 11](#_Toc200622284)

[DATE FUNCTIONS 12](#_Toc200622285)

[CONVERSION FUNCTIONS 12](#_Toc200622286)

[MULTIPLE ROW FUNCTIONS(GROUP FUNCTIONS) 13](#_Toc200622287)

[JOINS 13](#_Toc200622288)

[EQUI-JOIN OR INNER JOIN 14](#_Toc200622289)

[OUTER JOIN 14](#_Toc200622290)

DATABASE

# WHY DATABASE?

1. Organisation: Database neatly organise information allowing us to store large amount of data in one place.
2. Efficiency: With a database, we can search for any piece of data in seconds, saving time and effort.
3. Data Management: Databases allow easy updates, deletions or additions of data in an organised manner, reducing the risk of mistakes.
4. Security features ensuring that only authorised individuals can access the information keeping data safe.
5. Backup and Recovery: Data bases have backup recovery features that help restore data in case of failure.
6. Multiuser access: They enable multiple users to work on the same data simultaneously without interference.
7. Data Integrity: Databases ensure rule, ensure that data remains accurate and consistent across all operations.

# WHAT IS A DATABASE?

Database is an organised collection of interrelated data stored together without harmful or unnecessary redundancy.

# CHARECTERISTICS OF A GOOD DATABASE

1. PERFORMANCE: Facility for the retrieval and the manipulation of data irrespective of the number of tables with minimum time.
2. MINIMAL REDUNDANCY: The database should support minimal redundancy of data.
3. MULTIUSER: The Database should provide multi-user support
4. INTEGRITY: When multiple users use the Database, the data items and the associations between the data should not be destroyed
5. PRIVACY AND SECURITY: the data should be protected against accidental or intentional access by unauthorised persons.
6. THE DB LANGUAGE: The DB language used should be easy and powerful.

07-06-2025

# DATA MODELS

Data models are collection of tools for describing data and their inter relationships. They are:

1. OBJECT BASED LOGICAL MODEL
2. RECORD BASED LOGICAL MODEL
3. PHYSICAL DATA MODEL

## OBJECT BASED LOGICAL MODEL

#### ER MODEL

Represents data in terms of entities and the relationships between them. Used during the design phase to visualise logical structures.

#### OBJECT ORIENTED MODEL

Represents data in terms of their object’s classes and relationships.

## RECORD BASED LOGICAL MODEL

Organizes data into fixed formats or records.

#### RELATIONAL MODEL

Data is stored in tables with rows and columns.

#### NETWORK MODEL

Data is stored as records with links.

#### HIERARCHICAL MODELS

Data is organised in a tree structure.

## PHYSICAL DATA MODEL

Describes how data is physically stored in database, focusing on implementation details such as storage on hardware and performance optimisation.

# RELATIONAL MODEL TERMINOLOGY

Tables: The basic storage structure of the database holding all necessary data about real-world entities.

ROW(TUPLES): Contain all the data about an item.

COLUMN(ATTRIBUTE): Contain a particular piece of data for all entities.

CARDINALITY: The number of rows in a table.

DEGREE: The number of columns in a table.

# DDL-DATA DEFENITION LANGUAGE

## WHY DDL?

To store the data, first the structure to hold the data should first be created.

DDL is used to create, modify even remove the objects in the data base.

These statements define the structure of the database. It consists of statements that can create modify and remove database objects.

* CREATE-To create database objects
* ALTER- To change the structure of existing objects.
* DROP-To remove existing objects.
* TRUNCATE-To remove the data from the table.

# DATA TYPE

Each attributes holds a specific type of data.

Varchar(size): It’s a variable length character string having minimum length 1 and maximum length 4000.

Char(size): Fixed length character string with minimum length 1 and maximum length 2000.

Number(p,s): To store numbers

p🡪 Precision (Total number of digits)

s🡪Scale (Number of digits to right of decimal)

Date: To store date.

Long: Variable length character string upto 2 GB.

# ALTER

* ALTER-MODIFY:

Change the datatype of a column/increase /decrease the length of a column

ALTER TABLE TABLE\_NAME MODIFY COLUMN\_NAME NEW\_DATA\_TYPE

### ALTER ADD

Add new column to the existing table

ALTER TABLE TABLE\_NAME ADD NEW\_COLUMN\_NAME DATA\_TYPE

### ALTER RENAME

To change the name of a column

alter table TABLE\_NAME rename COLUMN OLD\_COLUMN\_NAME TO NEW\_COLUMN\_NAME

### ALTER DROP

To delete a column name from an existing table

ALTER TABLE TABLE\_NAME DROP COLUMN COLUMN\_NAME

# Problem

Department

Department id int

Name char(20)

Rating int

Location varchar(20)

1. Create the table given above
2. Change the datatype of name to varchar(20)
3. Change the length of location to varchar(50)
4. Add a new column ContactNo
5. Change the column name Location to Address

**RENAME**

To change the table name

RENAME OLD\_TABLE\_NAME TO NEW\_TABLE\_NAME

### **TRUNCATE:**

Deletes old records from the table

TRUNCATE TABLE TABLE\_NAME

09-06-2025

# CONSTRAINTS

We may need to apply certain restrictions on the value in a column.

* The values in a column should be unique.
* The values are mandatory
* Null not allowed.
* The value in the column should be greater than a specific value.
* Certain columns will be used for establishing relationships between tables.

# WHAT IS A CONSTRAINT

Used to define an integrity constraint a rule that restrain the value of one or more columns in a table.

It is applicable to create and alter statements.

* Not Null
* Unique
* Check
* primary key
* foreign key
* on delete cascade
* on delete set null

## Not Null

Create table Department(

Department\_id int not null,

Name varchar(20)

)

Restricts the entries of null but it allows duplicate values.

## Unique

Create table Department(

Department\_id int unique,

Name varchar(20)

)

It restricts duplicate values but it accepts null.

## Check

Create table Department(

Department\_id int not null,

Name varchar(20),

Salary Number(10,2) check salary(salary>10000)

)

To check values with user defined condition on a column.

## primary key

Create table Employee(

employee\_id int not null,

EmployeeName varchar(20),

contactNumber int,

CONSTRAINT emp\_pk primary key(employee\_id)

)

General Syntax:

CONSTRAINT CONSTRAINT\_NAME primary key(COLUMN\_NAME)

OR

ALTER TABLE TABLE\_NAME ADD CONSTRAINT CONSTRAINT\_NAME PRIMARY\_KEY(COLUMN)

Uniquely identifies each row in a table.

## foreign key

Create table Department(

Department\_id int,

Name varchar(20),

employee\_id int,

Constraint fk\_empid foreign key(employee\_id) references Employee(employee\_id)

)

General Syntax:

CONSTRAINT Constraint\_Name FOREIGN KEY(Foreign\_Key\_Name) REFERENCES Foreign\_Table\_Name(Foreign\_Key\_Name)

### on delete cascade

### on delete set null

# DML-DATA MANIPULATION LANGUAGE

DML commands are used to modify data in a database.

## INSERT

Adds new data to a table.

Insert all statement allows us to insert multiple rows into a table in a single query.

## UPDATE

Modifies existing data in a table.

UPDATE TABLENAME SET COLUMNNAME=NEWVALUE WHERE CONDITION

## DELETE

Removes data from a table

Delete statement removes one or more rows from a table based on a condition.

DELETE FROM TABLENAME WHERE CONDITION

11-06-2025

# FUNCTIONS

1. Single row functions
2. Multiple row functions

## Single row functions

1. Numeric functions
2. String functions
3. Date functions
4. Conversion functions

### NUMERIC FUNCTIONS

1. Abs() returns the absolute value

Select abs(-12) from dual

1. Ceil() returns a value that is greater than or equal to the given value.

Select ceil(9.3) from dual

1. Floor() returns the largest integer equal to or less than the given value

Select floor(9.8) from dual

1. Mod() used to return the reminder value

Select modd(10,2) from dual

1. Power() used to return the power of a given expression

Select power(2,3) from dual

### STRING FUNCTIONS

1. Length()

Select length(‘hello’) from dual

1. Lower()

select lower({“hello”) from dual

1. Upper()

Select upper(‘hello’) from dual

1. Initcap()

Select initcap(‘hello’) from dual

1. Ltrim() used to remove unwanted space or unwanted characters from the left side of the given string

Select ltrim(‘ aihan’) from dual

1. Rtrim() used to remove space or unwanted charecters from rightside of the given string

Select rtrim(‘aihan ’) from dual

1. Lpad()

Select lpad(‘hello,10, ‘+’) from dual

1. rpad()

select rpad(‘Hello’,10,’+’) from dual

1. concat()

select concat(‘computer’,‘science’) from dual

1. replace()

select replace(‘Hello world’,’world’,’universe’) from dual

1. select SUBSTR(‘ORACLE DATABASE’,1,6) FROM DUAL
2. SELECT INSTR(‘ORACLE DATABASE’, ‘d’) FROM DUAL

### DATE FUNCTIONS

1. Sysdate: Returns Current Date Of The Operating System On Which The Database Resides
2. ADD\_MONTHS: RETURNS THE DATE WITH A GIVEN NUMBER OF MONTHS ADDED

SELECT ADD\_MONTHS(SYSDATE,2) FROM DUAL

1. LAST DAY: RETURNS THE LAST OF THE MONTH THAT CONTAINS A DATE.

SELECT LAST\_DAY(SYSDATE) FROM DUAL

1. NEXTDAY: RETURNS THE DATE OF FIRSTWEEK DAY THAT IS LATER THAN THE DAY.

SELCET NEXT\_DAY(SYSDATE, ‘MONDAY’) FROM DUAL

1. MONTHS\_BETWEEN: TO GE THE NUMBER OF MONTHS BETWEEN TWO DATES

SELECT MONTHS\_BETWEEN(‘05/03/2025’, SYSDATE) FROM DUAL

1. CURRENT\_DATE: RETURN THE CURRENT DATE

SELECT CURRENT\_DATE FROM DUAL

### CONVERSION FUNCTIONS

* + TO\_CHAR: It converts the value of any datatype to string.

SELECT TO\_CHAR(SYSDATE, ‘YYY-MM-DD’) FROM DUAL

* + TO\_DATE: Converts a string to a date

SELECT TO\_DATE(‘02/03/1998’) FROM DUAL

* + TO\_NUMBER: Converts a string to number

## MULTIPLE ROW FUNCTIONS(GROUP FUNCTIONS)

* SUM()
* MAX()
* MIN()
* COUNT()
* AVG()

12-06-2025

# JOINS

In the database, the data needed for an application is distributed among different tables.

When we need to retrieve the data back in the form of a comprehensive report, we may need to take the data from two or more tables so as to get a full picture.

It is the concept of joins that enable us to implement this multi-table retrieval. The join is a keyword that helps to retrieve data from more than one table.

Depending on the type of output that we need and the joint condition there are different types of joins.

* Equi-join or Inner join
* Outer join
* Cross join
* Natural join
* Self join

## EQUI-JOIN OR INNER JOIN

The join that is used to retrieve details from the table and the corresponding data from another, only when there is a match in the join condition and the join condition involves an equality check is called Equi-Join.

## OUTER JOIN

The join that is used when we need all the rows from one table and the corresponding rows from the second table regardless whether there is a match or not.

For rows in the first table that has corresponding data in the second table, the data is shown. And for rows that do not have a report; NULL is displayed.

There are different types of outer join:

* Left outer join
* Right outer join
* Full outer join

# Self Study

* Self join
* Natural join
* NVL
* NVL2
* Coalesce

# SUB-QUERY

If a query needs to work based on the output of another query, then we use the subquery.

## Execution process

InnermostQuery to OutermostQuery

A single row subquery reurns one row and one column. It is typically used with comparison operators such as >,<,=,>=,=<,<>.

Multiple row subquery returns more than one row. It is used with operaytors like in,any,all,or exist

In : checks if value matches any value in a list or subquery

ANY: compares a value to each value in a list or subquery and returns true if any comparison is true.

ALL: compares a value to each value in a list or subquery and returns value in a list or subquery and returns true if all comparisons are true.

Exists: Checks if a subquery returns any rows.

13-06-2025

# VIEWS

View is a virtual table based on the result of a select query

It doesn’t store the data itself but provides a way to represent data stored in one or more table or other views.

Views can be used to simplify complex queries, encapsulate business logic and restrict access to certain columns or rows.

## CREATING A VIEW

CREATE VIEW view\_name AS

SELECT column1, column2…

FROM table\_name

WHERE CONDITION

## WHY VIEWS

The admin may need to restrict access to some data in the database.

## ADVANTAGES

* + Views enable us to view or present data in a different format than it appears on the memory
  + Views are subset of one or more tables.
  + View is referenced to as a virtual table as it doesn’t have data of its own.
  + View do not take up physical space in the database as tables do.
  + Whenever the view is accessed, the query is evaluated.
  + Changes in the views affect the table on which the view is based.
  + View help hide the ownership of the table and complexity of the query used to retrieve data from the user.
  + If we use order by clause in the view, it becomes a read-only view automatically.

## RESTRICTIONS ON VIEWS

* Updating or inserting is not possible if view is based on two tables.
* Insertion is not allowed if the underlined table has any NOT NULL columns that do not appear in the view.
* Insertion/updation is not allowed if any of the view referred in update or insert contains functions or calculation.
* Insertion/updation/deletion is not allowed if view contains DISTINCT clause.

## SYNTAX FOR VIEW BASED ON MULTIPLE TABLE

CREATE VIEW view\_name AS

SELECT columns1 FROM table1

JOIN table2

ON table1.common\_column=table2.common\_column

# SEQUENCE

Used to create a series of unique value.

Can be used for primary key attributes.

Can be used to create a sequence of values with specific increments.

## GENERAL SYNTAX

CREATE SEQUENCE SEQ\_NAME

INCREMENT BY N

START WITH N

MAXVALUE N|NOMAXVALUE

MINVALUE N|NOMINVALUE

# SYNONYMS

Synonyms are an alternative name for objects such as tables views sequence and other database objects.

Synonyms make it easier to access database objects by providing a shorter or mpre user-firendly name.

## WHY SYNONYMS

Instead of typing long object name, we can use a short synonym.

We can hide the underlyinhg schema from users by giving them access to synonyms instead of actual objects.

## TYPES OF SYNONYMS

* Public synonyms: available to all users of database.
* Private synonym: only available to the user who created it.

## SYNTAX

### PRIVATE SYNONYM

This is a default synonym type and is available to the user who created it.

CREATE SYNONYM SYNONYM\_NAME FOR OBJECT\_NAME

### PUBLIC SYNONYM

This allows all database users to access the object using the synonym.

CREATE PUBLIC SYNONYM SYNONYM\_NAME FOR OBJECT\_NAME

# NORMALIZATION

Normalisation is the efficiently organising data in database so that the database is free of unnecessary redundancy and anomalies - insert, update and delete anomalies - that leads to loss of data integrity.