SQL AND PYTHON : TOOLS IN CLOUD COMPUTING

Project report submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

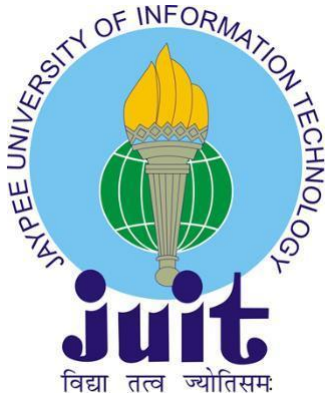
By

MOHIT GARG

161007

UNDER THE GUIDANCE OF

MANOPRINCY J
COGNIZANT TECHNOLOGY SOLUTIONS



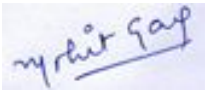
JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

MAY-JUNE 2020

TABLE OF CONTENT

DECLARATION

This is to certify that the work titled **"SQL and Python : Tools in Cloud Computing"**, submitted by **Mohit Garg** under the partial fulfillment for the award of Bachelor of Technology in Electronics and Communication of **Jaypee University Of Information Technology** has been carried out under my supervision. This work has never been submitted elsewhere for the award of this or any other degree or diploma.



Mohit Garg
161007

ACKNOWLEDGEMENT

I take this opportunity to express my sincere gratitude to the Prof. M. J. Nigam (HoD), Department of Electronics and Communication Engineering, Jaypee University of Information Technology, Solan for providing this opportunity to carry out the present work.

The constant guidance and consolation got from **Dr Shruti Jain** Associate Professor Department of Electronics and Communication Engineering has been of incredible assistance in conveying our present work and helped us in finishing this task with success.

I would like to express a deep sense of gratitude to my trainers **Mrs. Dipti Anjelakar** (SQL), **Mr. Krishanu Mukherjee** (JAVA), **Mr. Prabhat Chandra** (Python), at Cognizant Technology Solutions for their guidance and support in defining the design problem and towards the completion of my work. Without their wise counsel and able guidance, it would have been impossible to complete the thesis in this manner.

I am also thankful to our batch owners my batch owner **Mrs. Manoprincy J,** for her intellectual support throughout the course of this work.

Finally, I thank my parents and my family members for giving the moral support and abundant blessings in all of activities and my dear friends who helped me to endure the difficult times with their unfailing support and warm wishes.

LIST OF TABLES

LIST OF FIGURE

ABSTRACT

With increasing amount of data and services enabling us to use computing across various products, cloud computing has emerged as an exciting paradigm. It helps in sharing resources as well as information between different devices . DBMS or database management system acts through cloud computing platforms like AWS or Amazon Web Services. With time , involvement of DBMS in the sphere of cloud computing is going to enhance ridiculously. With DBMS we are able to execute queries , build indexes and load different types of data . Through SQL or standard query language, implementation of DBMS concepts becomes very fruitful.

A clear understanding for Java-based web apps is necessary to run them on cloud components so that clients can access cloud provided softwares .With clear understanding of OOPs based concepts like encapsulation, inheritance , polymorphism etc. the base and foundation of core developing is enhanced.

Python has come out as one of the most powerful language developers want to use to develop codes and softwares. It's application in cloud computing is promising.

It powers some of the most string applications on cloud.

This report consist of basic foundation of Python which has helped in analysing and organising data for implementation. It is especially useful for streaming analytical applications built on the cloud.

For the pivotal growth of cloud computing across various platforms , it has become important to collect information on cloud and analyse them and find out useful information and for that purpose Python is crucial. Using a combination of Python -based analytics, cloud data services and visualization tools, we are able to make innovative applications around the cloud platform .

With basic foundation of SQL in DBMS, JAVA as Object oriented programming language and Python , the works for cloud computing becomes very wide.

CHAPTER 1 DBMS AND SQL

INTRODUCTION

1.1 WHAT IS DATA?

An object of interest can be designated as data. Any information which we can collect about these interest can be called as data.e.g. data about a student would include information like name, address, age, semester grades etc. Software Applications store data to answer a question e.g. how many students have a gpa greater than 8 .Data can be used to convey many information about any entity.
Within itself data is just a raw representation. When this raw information is used to find some relevant patterns then it is called as information. This information in turn when used to make basic decision then it is called knowledge.

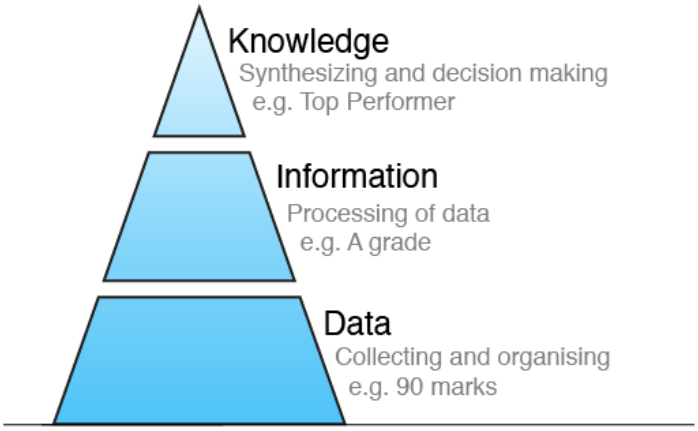


Figure 1. 1 Data Hierarchy

1.2 REQUIREMENTS FROM DATA

End users expect some end result from software applications. Let us take the example of Facebook application.

Table 1. 1 Requirements from Data

Requirement	Description
Integrity	Data should be accurate e.g. my facebook profile should contain valid country name.
Availability	I should be able to access facebook and see my data at all times.
Security	Only my friends should be able to see my posts and no one else.
Independent of Application	my Android app as well as web browser on my laptop should be able to access data.
Concurrent	All my friends should be able to see my posts at the same time.

1.3 LIMITATIONS OF FLAT FILES

Data is stored in flat files and can be accessed using any programming language. The file based approach suffers following problems:

1. Dependency of program on physical structure of data
2. Complex process to retrieve data
3. Loss of data on concurrent access
4. Inability to give access based on record (Security)
5. Data redundancy

1.4 DATABASE SYSTEMS

A common assemblage which is designed to meet infp of any firm is related logically and such a representation is database .
A software system to control access and define or create a database can be done by a Database Management System. Database Systems demand a high hardware configuration and sometimes ann optimised cost.
An Application Program interacts with a database by issuing an appropriate request (typically a SQL statement).

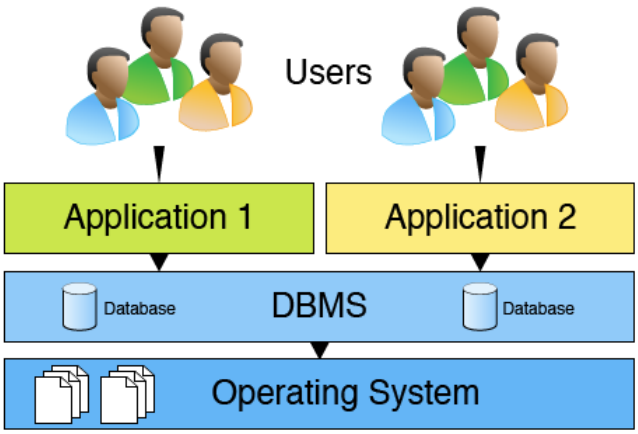


Figure 1. 2 Database System

1.5 FUNCTIONS OF DBMS

Database Management Systems offer several functions that help us overcome problems associated with file based systems.

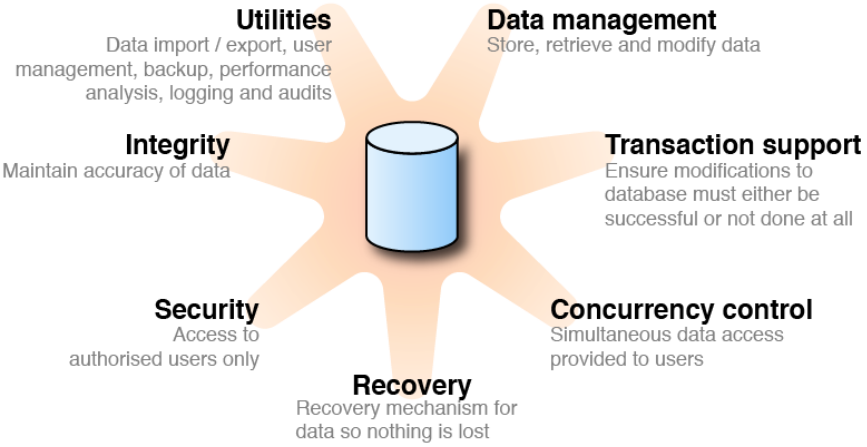


Figure 1. 3 Functions of DBMS

1.6 TYPES OF DATABASE SYSTEMS

These database systems can be classified into four types in terms of their chronology of evolution:hierarchical, network,Relational ad NoSql. We will now get a brief overview of these database management systems.

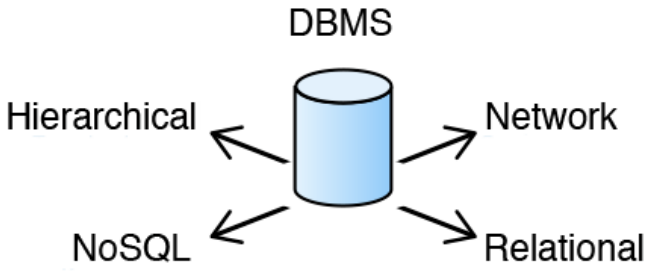


Figure 1. 4 Types of Database Systems

Hierarchical Databases sort out information into a tree-like structure. Information is put away as records which are associated with each other through parent kid connections. A few instances of Hierarchical Databases are Information Management System (IMS), Raima Database Manager (RDM) Mobile and so forth.

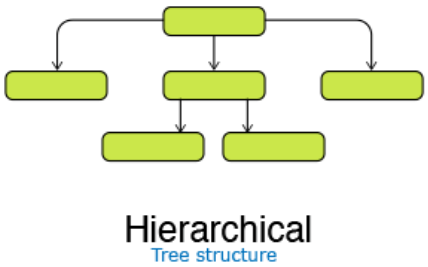


Figure 1. 5 Hierarchical Database

Network Databases compose information into a diagram structure in which item types are hubs and relationship types are curves. Each record can have various parent and youngster records. A few instances of Network Databases are Integrated Database Management System (IDMS), Integrated Data Store (IDS) and so forth.

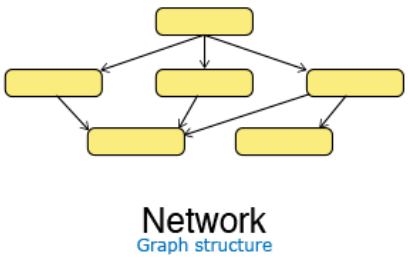


Figure 1. 6 Network Database

Relational Databases sorts out information into at least one tables. A table comprises of qualities (segments), tuples (pushes) and gives an approach to particularly distinguish each tuple. Tables are identified with one another through parent youngster connections. A few instances of Relational Databases are DB2, Oracle, SQL Server and so forth.

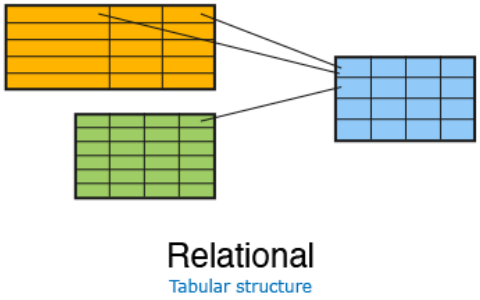


Figure 1. 7 Relational Database

NoSQL (Not only SQL) database uses key-value, graph or document data structures to store data. These databases aim for simplicity of design, horizontal scaling and finer control over availability. Some examples on No Sql databases are Cassandra, MongoDB, CouchDB, OrientDB, HBASE etc.

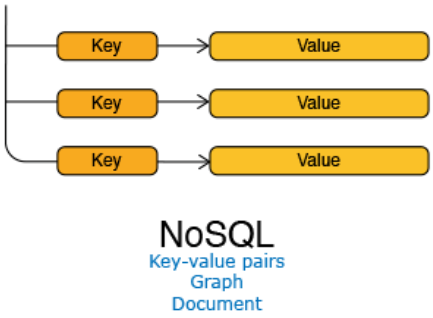


Figure 1. 8 No SQL Database

1.7 DATA INTEGRITY AND CONSTRAINTS

Data integrity alludes to keeping up and guaranteeing the exactness and consistency of information over its whole life-cycle. Database Systems guarantee information honesty through requirements which are utilized to confine information that can be entered or altered in the database. Database Systems offer three types of integrity constraints:

Table 1. 2 Integrity Types

Integrity Types	**Definitions**	**Enforced Through**
Entity Integrity	Each table must have a column or a set of columns through which	PRIMARY KEY

	we can uniquely identify a row. These column(s) cannot have empty (null) values.	
Domain Integrity	All attributes in a table must have a defined domain i.e. a finite set of values which have to be used. When we assign a data type to a column we limit the values that it can contain. In addition we can also have value restriction as per business rules e.g. Gender must be M or F.	CHECK CONSTRAINT, DATA TYPES
Referential Integrity	Every value of a column in a table must exist as a value of another column in a different (or the same) table.	FOREIGN KEY

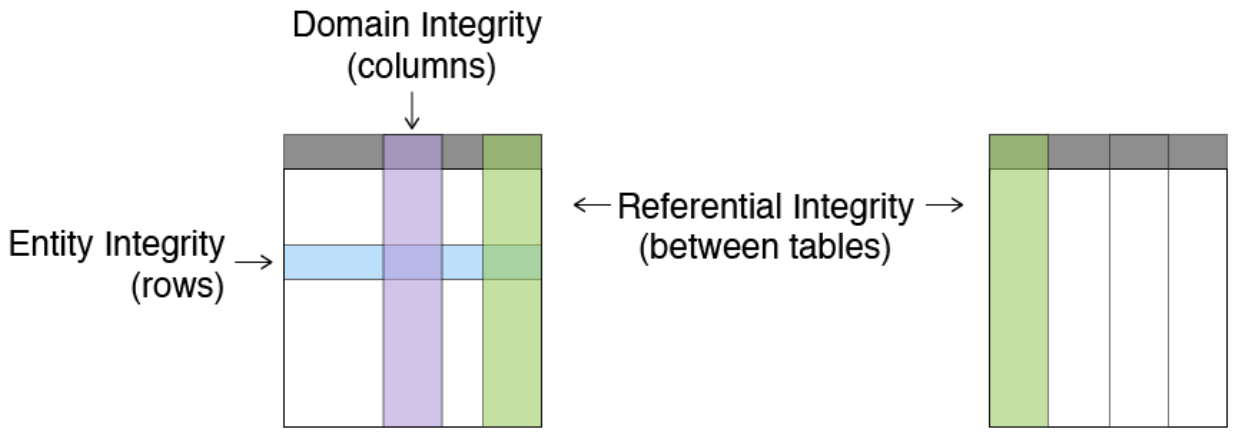


Figure 1. 9 Data Integrity

CANDIDATE KEY:- A minimal set of columns that can uniquely identify a single tuple in a relation. What underlying business decisions are to be taken through database , that determines what candidate keys would comprise of.

PRIMARY KEY:- To uniquely identify a tuples in a database we require primary key.

Table 1. 3 Mandatory and Desired attributes for a primary key

Mandatory	Desired
must uniquely identify a tuple	should not change with time
must not allow NULL values	should have short size e.g. numeric data types

When two or more columns together identify the unique row then it's referred to as Composite Primary Key.

FOREIGN KEY:- We have two tables , one primary table , one secondary table. We select one key from primary table which can uniquely identify the tupeles in the secondary table. And this key is called foreign key. A relationship is formed as soon as a foreign key is formed between two tables. Let us take Employee and Computer tables as provided below:
Computer is the primary table with CompId as the primary key. Employee is the secondary table with Id as the primary key.

Parent / master / referenced table

Computer table

COMPID	MAKE	MODEL
1001	Dell	Vostro
1002	Dell	Precision
1003	Lenovo	Edge

Child / referencing table

Employee table

ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	1002

Figure 1. 10 Foreign Key

1.8 ENTITY RELATIONSHIP (ER) MODEL

ER model is used to graphically represent entities and relationship helping to understand data independency of the actual database implementation. Let us understand some key terms used in ER Modelling.

Table 1. 4 ER Model Terms

Term	Definition	Examples
Entity	Real world objects which have an independent existence and about which we intend to collect data.	Employee, Computer
Attribute	A property that describes an entity.	Name, Salary

A sample ER Diagram representing the Employee entity along with its attributes is presented below:

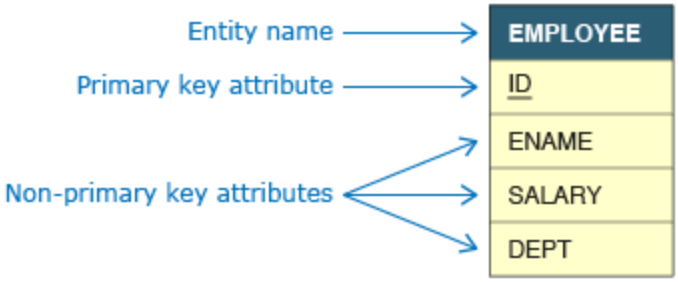


Figure 1. 11 ER Model

Crow foot notation is one of the ways to represent cardinality of relationship in an ER Model. The notation comprises of four symbols and one of them need to be used for each entity in a relationship.

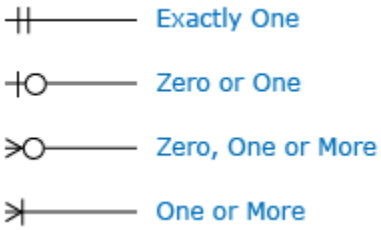


Figure 1. 12 Cross Foot Notation

1.9 SQL

Structured Query Language (SQL) used to handle data that is stored in various types of databases.We find 4 different types of commands in SQL which have different types of functions useful in sql.

1. **DDL-Data Definition Language**:A relational database has a specific schema which can be described by DDL commands.There are various types entities , indexes, constraints and keys in a database and they can be created , modified and deleted which is done by DDL commands.

2. **DML-Data Manipulation Language**:eSometimes when we have created a database , we need to update some values or alter it or modify it according to the current needs and this is where DML comes into play.It hides the specific implementation and still provides what the user asked for.

3. **DCL-Data Control Language:** views, tables, stored procedures etc. in a relational database can be accessed through DCL.It grants and revokes privileges of a database.

4. **TCL-Transaction Control Language**:to begin or end a transaction in a database we need TCL commands. a sequence of SQL statements comprises a transaction that are applied in an atomic (all or none) manner. A commit makes all the changes applied by the transaction permanent on the database while a rollback undoes all the changed applied by the transaction.

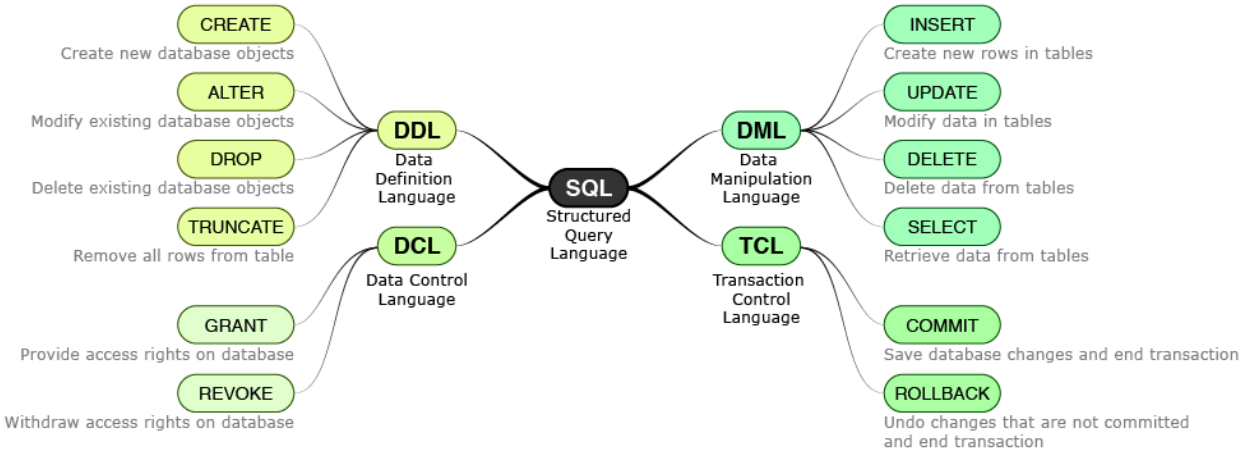


Figure 1. 13 Parts of SQL

1.9.1 DATA TYPES

There are three type of data types that SQL supports namely:

1. **SQL Character Data Types**: SQL supports two character data types for storing printable and displayable characters. They are used for storing information like name, address, description etc.

2. **SQL Integral Data Types**: SQL supports SMALLINT, INTEGER and INT data types that are used for storing whole numbers. Unlike other databases Oracle does not define different size limits for them. They are all treated internally to have 38 digit of precision.

3. **SQL Non-Integral Data Types**: Nonintegral data types have an integer part and a fractional part. Either NUMERIC, DECIMAL or NUMBER data types can be used to store nonintegral numbers

1.9.2 DDL STATEMENTS

1. CREATE TABLE :to create a table in a database. Rows and columns are created . Each table has to compulsorily have a name and can have any number of columns (minimum 1 column is required). Each column must have a data type which determines the type of values that can be stored. CREATE TABLE command will fail if a table is already existing in the database with same name. All tables must have a unique name.

   SYNTAX:
   ```
   CREATE TABLE Student (
   StudentId INTEGER,
    FName VARCHAR2(10),
   Gender CHAR(1),
     DOJ DATE);
   ```

2. DROP TABLE removes an existing table from the database.

   SYNTAX-DROP TABLE Employee;

CONSTRAINTS

Constraints are typically specified along with the CREATE TABLE statement. Constraints are classified into multiple types based on the number of columns they act upon as well as on the way they are specified. Various constraints that can be created on database tables are:

1. **NOT NULL CONSTRAINT**: NOT NULLdoes not let a column from accepting NULL values. NOT NULL can only be applied as a column level constraint. Constraint name is optional and it can be specified by using CONSTRAINT keyword.

   SYNTAX: CREATE TABLE Employee (

   EMpId INTEGER CONSTRAINT tud_EId_nn NOT NULL,
   FullName VARCHAR2(30) NOT NULL,
   LastName VARCHAR2(30)

2. **DEFAULT CONSTRAINT**: A column can be given the default value by using DEFAULT option. The data type of column and default expression must be the same. DEFAULT option can be provided for nullable as well as NOT NULL attributes. Oracle database does not consider DEFAULT as a constraint.

   SYNTAX: CREATE TABLE Student (
   StudentId INTEGER,
   FName VARCHAR2(10),
   DOJ DATE DEFAULT SYSDATE);

3. **PRIMARY KEY CONSTRAINT**: This constraint on a column ensures that the column cannot contain NULL and duplicate values.

   SYNTAX- CREATE TABLE Student (
   StudentId INTEGER CONSTRAINT stud_sid_pk PRIMARY KEY,
   FName VARCHAR2(10),
   ContactNo NUMBER(10));

4. **CHECK CONSTRAINT**: To limit the values that can be specified for a column.

   SYNTAX: CREATE TABLE Student (
   StudentId INTEGER,
   FName VARCHAR2(10),

   Gender CHAR(1) CONSTRAINT Stud_gender_ck1 CHECK(Gender IN('M', 'F')));

5. **UNIQUE CONSTRAINT**: To ensure that two rows in a table cannnot have same value in that column. Unlike Primary Key, UNIQUE constraint allows NULL values. many UNIQUE constraints can be present in one table.
   SYNTAX: CREATE TABLE Student (
   StudentId INTEGER, FName VARCHAR2(10), ContactNo NUMBER(10) CONSTRAINT Stud_cno_uk UNIQUE);

1.9.3 OPERATORS:

We have many types of operations possible in SQL:

Arithmetic operators:

Table 1. 5 Arithmetic Operators

Operator	Symbol	Usage	Result
Addition	+	15 + 5	20
Subtraction	-	15 - 5	10
Multiplication	*	15 * 5	75
Division	/	15 / 5	3

Comparison Operator:

Table 1. 6 Comparison Operators

Operator	Symbol	Usage	Result
Equal to	=	15 = 5	false
Not equal to	<>	15 <> 5	true
Greater than	>	15 > 5	true
Greater than equal to	>=	15 >= 5	true
Less than	<	15 < 5	false
Less than equal to	<=	15 <= 5	false

Operator	Symbol	Usage	Example
Range	BETWEEN *<lower limit>* AND *<upper limit>*	Matches value between a range of values (Both inclusive)	Salary BETWEEN 2500 AND 3000
List	IN (List of values)	Matches any of a list of values	Dept IN ('IVS', 'ETA', 'ICP')
String pattern matching	LIKE	Matches a character pattern	SupplierId LIKE 'S%'
NULL Test	IS NULL	Is a null value	Bonus IS NULL

Logical Operator:

Table 1. 7 Logical Operators

Operator	Symbol	Usage	Example
And	AND	Returns TRUE if both conditions are true	Salary >= 30000 AND Dept = 'ETA'
Or	OR	Returns TRUE if any one of the condition is true	Salary > 75000 OR Dept = 'ICP'
Not	NOT	Returns TRUE if following condition is false	Id NOT IN (2,3)

1.9.4 FUNCTIONS:

Conversion Functions:

Table 1. 8 Conversion Function

Name	Syntax	Function
TO_CHAR	TO_CHAR(value, format)	Converts a number or a date to a string. Use this function for formatting dates and numbers.
TO_DATE	TO_DATE (value, format)	Converts a string to a date.
TO_NUMBER	TO_NUMBER (value, format)	Converts a string to a number.

Date Function:

Table 1. 9 Date Function

Name	Syntax	Function
SYSDATE	SYSDATE	Returns current date of System i.e. the host on which database server is installed.
SYSTIMESTAMP	SYSTIMESTAMP	Returns current timestamp of the System.
ADD_MONTHS	ADD_MONTHS(date, n)	Adds n months to the given date.
MONTHS_BETWEEN	MONTHS_BETWEEN(date1,date2)	Finds difference between two dates in months.

CASE STATEMENT

It is used to select from a range of values depending upon the condition. We can use WHERE, GROUP BY with case statement . We have following Examples:

1. Simple CASE expression:

SQL CASE statement

```
CASE Designation
    WHEN 'SE' THEN Salary * 1.2
    WHEN 'SSE' THEN Salary * 1.1
    ELSE Salary * 1.05
END
```

Figure 1. 14 Simple CASE expression

2. Searched CASE expression:

SQL CASE statement

```
CASE
    WHEN Marks >= 85 THEN 'Excellent'
    WHEN Marks >= 65 THEN 'Good'
    ELSE 'Poor'
END
```

Figure 1. 15 Searched CASE expression

ALTER TABLE

We need to use ALTER TABLE command through which the structure of existing table can be changed without any loss of data. ALTER table can also be used to rename a column, change data type of a column and add or remove constraints. Syntax for ALTER TABLE command is provided below:

Clauses

Alter statement 1	`ALTER TABLE Student`	`ADD Address VARCHAR2(20)`
Alter statement 2	`ALTER TABLE Student`	`MODIFY Address VARCHAR2(50)`
Alter statement 3	`ALTER TABLE Student`	`RENAME COLUMN Address TO ResidentialAddress`
Alter statement 4	`ALTER TABLE Student`	`DROP (ResidentialAddress)`
Alter statement 5	`ALTER TABLE Student`	`ADD CONSTRAINT stud_sid_pk PRIMARY KEY (StudentId)`
Alter statement 6	`ALTER TABLE Student`	`DROP CONSTRAINT stud_sid_pk`

Figure 1. 16 ALTER STATEMENT

INSERTING DATA

INSERT STATEMENT: To add tuples (records) to table. It supports many syntax as shown below:

Table Name

INSERT statement 1

```
INSERT INTO Employee
VALUES
    (1,'James Potter','01-Jun-2014', 40000.00, NULL,'FSI','TA', NULL, 1)
```

Figure 1. 17 INSERT STATEMENT

RETRIEVING DATA

SELECT QUERY: SELECT query allows us to retrieve data from many tables in a relational database. It can be represented as:

Attributes — Table name (Relation)

Select query 1	`SELECT *`	`FROM Employee`	
Select query 2	`SELECT Id, EName`	`FROM Employee`	
Select query 3	`SELECT Id, EName`	`FROM Employee`	`WHERE Salary > 40000`

Required clauses

Optional WHERE clause with conditions for selecting data

Figure 1. 18 SELECT STATEMENT

Use **DISTINCT** clause to remove duplicates. Usage of DISTINCT should be avoided as far as possible as it can lead to performance issues.

SYNTAX: SELECT DISTINCT Dept FROM Employee

WHERE CLAUSE

To refine records on a basis of a condition we need WHERE statement. Only those data that meet some required conditions get filtered with this clause.

SYNTAX:
```
SELECT col1, col2, ...
FROM name_of_table
WHERE condition;
```

LIKE OPERTOR

LIKE operator is used to match a character pattern. It allows us to use wild cards. SQL supports two wild cards: '%' which matches with any number of characters and '_' which matches with exactly one character.

SYNTAX-
```
SELECT col1, col2, ...
FROM name_of_table
WHERE col LIKE some_pattern;
```

UPDATING DATA

UPDATE SYNTAX

To modify records existing inside a single table in a database. Update statement can be represented as:

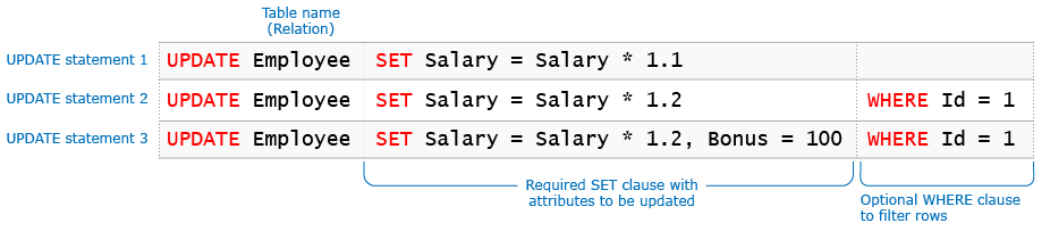


Figure 1. 19 UPDATE STATEMENT

The database system ensures that no constraints are violated during execution of an update statement. Any violation of constraints results in failure of the statement.

DELETING DATA

DELETE : deletes records from a table in a relational database. The database system ensures that no constraints are violated during execution of a delete statement. Any violation of constraints results in failure of the statement.

TRUNCATE statement can also be used to delete data from tables. TRUNCATE statement deletes every rows from table as it does not support WHERE clause. TRUNCATE statement is a faster option compared to DELETE when we have to delete all rows from the table.

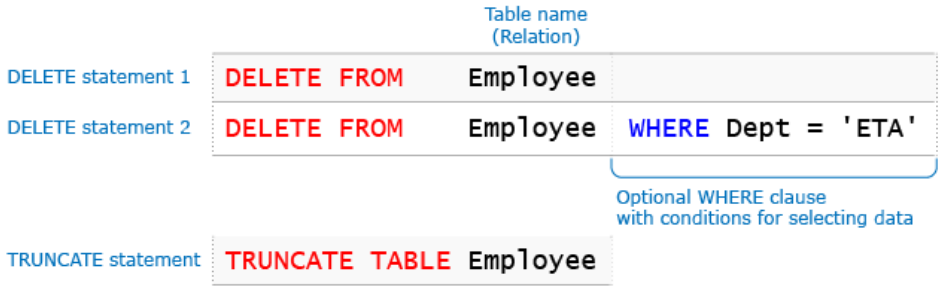


Figure 1. 20 DELETE AND TRUNCATE STATEMENT

1.9.5 SQL FUNCTIONS

SQL functions are built in modules provided by a database. We can use them in data manipulation statements to perform calculations on data. All functions return a single value.

NUMERIC FUNCTION

Numeric functions are single row functions that accept a numeric value and return numeric output.

Table 1. 10 Numeric Function

Name	Syntax	Function
ABS	ABS(value)	Returns absolute value of a number
ROUND	ROUND(value, digits)	Rounds the value to specified decimal digits
CEIL	CEIL(value)	Rounds up the fractional value to next integer
FLOOR	FLOOR(value)	Rounds down the fractional value to the lower integer

CHARACTER FUNCTION

Character functions work on character strings and can return a character string or a numeric value.

Table 1. 11 Character Function

Name	Syntax	Function
UPPER	UPPER(value)	Converts value to upper case
LOWER	LOWER(value)	Converts value n lower case
CONCAT	CONCAT(value1, value2)	Concatenates value1 and value2
LENGTH	LENGTH(value)	the number of characters in value present returned

SUBSTRING FUNCTION

Substring function is used to extract part of a string. It has the following syntax SUBSTR(value, start_position, length)

Figure 1. 21 Substring Function

AGGREGATE FUNCTION

Aggregate functions works on many rows at a time to return a single row. Some aggregate functions like SUM (total), AVG (average) operates only on numeric columns while others like MIN (lowest value), MAX (highest value) and COUNT (number of rows) operate on all data types. All aggregate functions ignore NULL values except COUNT(*).

SYNTAX: SELECT MIN(Salary), MAX(Salary), SUM(Salary) FROM Employee

SYNTAX: SELECT COUNT(ID) COUNT_ID, COUNT(*) COUNT_STAR, COUNT(Bonus) COUNT_BONUS FROM Employee

SYNTAX: SELECT AVG(Salary) AvgSalary, AVG(Bonus) AvgBonus1, SUM(Bonus) / Count(Bonus) AvgBonus2 FROM Employee

1.9.6 SORTING DATA

ORDER BY CLAUSE

Order By clause sorts the answer of a query in a particular order. Before we understand the syntax of ORDER BY, Sorting on secondary column happens only when multiple rows have the same value in the primary column. The sort order can be different for the two columns i.e. primary can be sorted in ascending and secondary in descending and vice-versa. ORDER BY must be used to specify the columns on which data has to be sorted and the sort order i.e. ascending or descending. Rows are sorted in ascending order if sort order is not specified. DESC should be used to sort the rows in descending order.

```
ORDER BY example 1   SELECT Id, EName, Salary FROM Employee ORDER BY EName
ORDER BY example 2   SELECT Id, EName, Salary FROM Employee ORDER BY EName, Salary
ORDER BY example 3   SELECT Id, EName, Salary FROM Employee ORDER BY EName DESC
ORDER BY example 4   SELECT Id, EName, Salary FROM Employee ORDER BY EName ASC, Salary DESC
ORDER BY example 5   SELECT Id, EName, Salary FROM Employee ORDER BY 2, 3
```

Figure 1. 22 ORDER BY CLAUSE

GROUPING DATA

GROUP BY

GROUP BY does grouping of the data from the table into various groups based on criteria provided and calculates the aggregate function for each group. Thus the result has 1 row for each group.

SYNTAX: SELECT Dept, Designation, MAX(Salary) FROM Employee GROUP BY Dept, Designation;

HAVING

Having allows aggregate functions to be used as filter criteria which cannot be done using WHERE clause.

COMBINING DATA

UNION and UNION ALL clause to combine results from two or more SELECT statements. The select statements may be from same or different tables.They must have same number of columns and their data types at same position in both the query must be compatible (either same or convertible through automatic conversion). UNION removes all duplicates from the result. Two records are considered duplicates if values at corresponding positions of all their columns match.

SYNTAX: SELECT CompId FROM Employee UNION SELECT CompId FROM Computer

1.9.7 JOIN CLAUSE

A JOIN clause combines rows from two or more tables, basis formed is the column present in them

CROSS JOIN:

It is referred to as cartesian product . When we have two table with m and n rows respecitivlely thne the cross k=join gives m * n rows . It gives a lot of meaningless data that is why not generally used.
Algorithm:

```
for each row r1 in Employee table
    for each row r2 in Computer table
        add combined row to Result
```

Figure 1. 23 Algoritm of Cross Join

Example:

Employee Table

ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	NULL

Figure 1. 24 Employee Table

Computer Table:

COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014

Figure 1. 25 Computer table

Now the cross join of Employee and Computer Table gives:

ID	ENAME	ECID	CCID	MODEL
1	James Potter	1001	1001	Vostro
1	James Potter	1001	1002	Precision
2	Ethan McCarty	NULL	1001	Vostro
2	Ethan McCarty	NULL	1002	Precision
3	Emily Rayner	NULL	1001	Vostro

Figure 1. 26 Cross Join

INNER JOIN

Records that have matching values in both tables are selected in inner join.

Algorithm:

```
for each row r1 in Employee table
    for each row r2 in Computer table
        if r1.COMPID == r2.COMPID
            add combined row to Result
```

Figure 1. 27 Algorithm of Inner Join

Employee Table

ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	1002

Figure 1. 28 Employee table

Computer Table

COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014

Figure 1. 29 Computer Table

Result of Inner Join

ID	ENAME	ECID	CCID	MODEL
1	James Potter	1001	1001	Vostro
3	Emily Rayner	1002	1002	Precision

Figure 1. 30 Inner Join

SYNTAX- SELECT col_names(s)
FROM tableA
NNER JOIN tableB
ON tableA.colA_name = tableB.colB_name;

LEFT JOIN

Returns all records from the left table (tableA), and the records that are matching from the right table (tableB).

Algorithm:

```
for each row r1 in Employee table
    set matched_in_comp to false
    for each row r2 in Computer table
        if r1.COMPID == r2.COMPID
            add combined row to Result
            set matched_in_comp to true
    if matched_in_comp is false
        add Employee row to Result
```

Figure 1. 31 Algorithm of Left Join

Employee Table:

ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	NULL

Figure 1. 32 Employee table

Computer Table:

COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014

Result

ID	ENAME	ECID	CCID	MODEL
1	James Potter	1001	1001	Vostro
2	Ethan McCarty	NULL	NULL	NULL
3	Emily Rayner	NULL	NULL	NULL

Figure 1. 33Computer Table and Left Join

SYNTAX- SELECT col_name(s)
FROM tableA
LEFT JOIN tableB
ON tableA.colA_name = tableB.col_name;

RIGHT JOIN

Returns all records from the right table (tableB), and the records that are matching from the left table (tableA).

Algorithm:

```
for each row r1 in Employee table
    for each row r2 in Computer table
        if r1.COMPID == r2.COMPID
            add combined row to Result
            set matched_in_emp to true
for each row r3 in Computer table
    if not matched_in_emp is true
    add computer row to Result
```

Figure 1. 34 Algorithm of Right Join

Employee Table:

ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	NULL
3	Emily Rayner	ETA	NULL

Figure 1. 35 Employee table

Computer Table:

COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014

Figure 1. 36 Computer Table

Result:

ID	ENAME	ECID	CCID	MODEL
1	James Potter	1001	1001	Vostro
NULL	NULL	NULL	1002	Precision

Figure 1. 37 Right Join

SYNTAX: SELECT col_name(s)
FROM tableA
RIGHT JOIN tableB
ON tableA.colA_name = tableB.colB_name;

FULL OUTER JOIN

Full outer join combines the result of both left join and right join.
Illustrative Algorithm:

```
for each row r1 in Employee table
    set matched_in_comp to false
    for each row r2 in Computer table
        if r1.COMPID == r2.COMPID
            add combined row to Result
            set matched_in_comp to true
            set matched_in_emp to true
        if matched_in_comp is false
            add employee row to Result
for each row r3 in Computer table
    if not matched_in_comp is true
        add computer row to Result
```

Figure 1. 38 Algorithm of Full Outer Join

Employee Table

ID	ENAME	DEPT	COMPID
1	James Potter	ICP	1001
2	Ethan McCarty	ETA	null
3	Emily Rayner	ETA	NULL

Figure 1. 39 Employee table

Computer Table

COMPID	MAKE	MODEL	MYEAR
1001	Dell	Vostro	2013
1002	Dell	Precision	2014

Figure 1. 40 Computer Table

Result:

ID	ENAME	ECID	CCID	MODEL
1	James Potter	1001	1001	Vostro
2	Ethan McCarty	null	null	null
3	Emily Rayner	null	null	null
null	null	null	1002	Precision

Figure 1. 41 Full Outer Join

1.9.8 SUBQUERY

Subquery is a query which is present within another query. A subquery must be enclosed in brackets and can be used in SELECT, FROM, WHERE and HAVING clauses. Subquery in SELECT and FROM clause are rarely used. Subqueries in WHERE and HAVING clauses are classified into Independent and Correlated subqueries.

INDEPENDENT SUBQUERY

In an independent subquery, the inner and outer query are independent of each other. We can run an inner query and inspect its result independent of the outer query. Independent subquery are further classified into single row and multiple row types depending upon the number of rows returned.

```
SELECT ID, EName, Salary FROM Employee A WHERE Salary
= (SELECT MAX(Salary) FROM Employee B)
```

Figure 1. 42 Independent Subquery

CORRELATED SUBQUERY

A Correlated subquery is one in which the inner query is dependent on the outer query for it's complete execution. Specifically it uses a column from one of the tables in the query which is outer. The inner query is executed iteratively for each selected row of the outer query. In case of independent subquery, the inner query just executes once.

Figure 1. 43 Correlated Subquery

1.10 SOFTWARE USED

MYSQL WORKBENCH

MySQL Workbench is an integrated virtualization tool for repositories, developers, and DBAs. MySQL Workbench provides data model, SQL development, and complete management tools for server configuration, user management, backup and much more. MySQL Workbench is available for Windows, Linux and Mac OS X. MySQL Workbench enables the DBA, developer, or data architect to visualize, model, generate, and manage databases. It includes all the data modeler required to build complex ER models, forwards and backs up engineering, and also brings important aspects of the change management task and complexity of time-consuming documentation and effort. MySQL Workbench delivers visual tools for performing, executing, and optimizing SQL queries. The SQL editor provides syntax color highlighting, auto-completion, reuse of SQL captions, and SQL execution history. The Database Connectivity panel enables developers to easily manage standard data communications, including MySQL Fabric. The Object browser provides quick access to the database schema and objects.

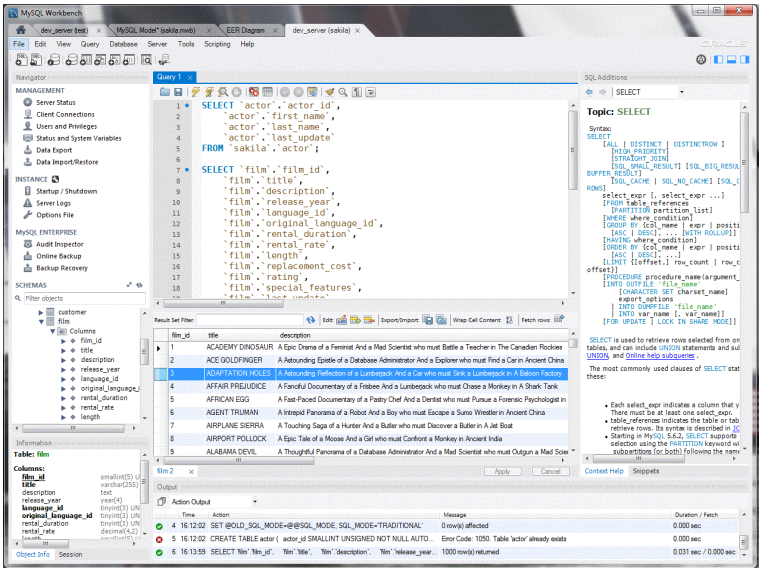


Figure 1. 44 MYSQL Workbench

CHAPTER 2 PROGRAMMING FUNDAMENTALS USING JAVA

2.1 INTRODUCTION

Need for Oops

1. It provides methodology for achieving benefits of reusable software components.
2. The methodology is quicker as it provides reusability features.
3. It places focus on object and information. A problem can be broken down into many objects .
4. It was created to overcome shortcomings of structured programming techniques.
5. It uses bottom up approach.
6. The foundation of the code is a class which is used to form run time objects.

Key features of Oops

1. Encapsulation
2. Abstraction
3. Inheritance
4. Polymorphism

How is JAVA platform independent ?

The significance of platform independent is that the java incorporated code(byte code) can run on every single working framework.

A program is written in a language that is a comprehensible language. It might contain words, phrases, and so forth which the machine doesn't comprehend. For the source code to be comprehended by the machine, it should be in a language comprehended by machines, normally a machine-level language. In this way, here comes the job of a compiler. The compiler changes over the significant level language (human language) into an organization comprehended by the machines. Along these lines, a compiler is a program that interprets the source code for another program from a programming language into executable code.

This executable code might be an arrangement of machine guidelines that can be executed by the CPU legitimately, or it might be a middle of the road portrayal that is deciphered by a virtual machine. This middle of the road portrayal in Java is the Java Byte Code.

Step by Step method of execution:

1. The javac compiles the code written in JAVA.
2. The result is .class file or the bytecode. And not machine code.
3. The bytecode which we have obtained is non executable. And an interpreter will be needed.
4. Finally program is run to give the output

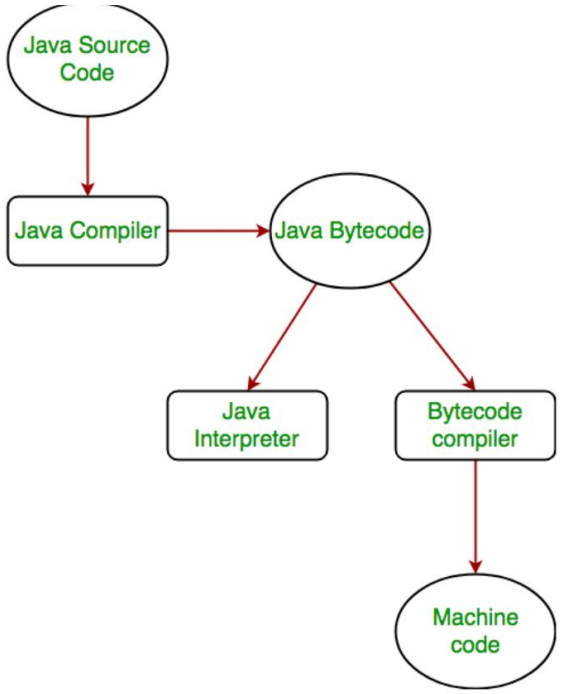


Figure 2. 1 Java

Java is platform independent but the JAVAC is platform dependent.

ENUM

ENUM: To represent a group of constants which are named in a language which is called Enum.
If we take an example of planets , We can have an enum named Solar System with enumerators called Earth, Mars, Venus, Jupiter. SImilarly we can have many enums like that of colors or directions, etc.
When all possible :
-Constructor(s) must have the same name as the class within which it is defined while it is not necessary for the method in java.
-Constructor(s) do not return any type while method(s) have the return type or void if does not return any value.
-Constructor is called only once at the time of Object creation while method(s) can be called any number of times.

1. values are to be known at compile time then we use enum.
2. The set of enum need not be fixed all the time.
3. We use enum keyword to declare enum which means it is of enum datatype.
4. The main objective which we have is to define our data types.

How to declare ?
We can do it either outside of a class or inside of a class but we cannot do it inside a method.

EXAMPLE: enum Direction
{
EAST,WEST,NORTH,SOUTH;
}
main()
{
Direction d=DIrection.EAST;

}

One thing to keep in mind is that inside enum that the first line should have a list of things like methods, constructors, variables , constants and other things.
According to Java conventions of naming, we should name the constraints with capital letters.

Some points to Enum:

1. Class internally implements enum.
2. Object of type enum are represented by enum constants.
3. Enum constant always implicitly public static final.

2.2 METHODS

SYNTAX:

Return type name ()

{

Body

}

We accompany them with modifiers such as default, private, public , private, protected and others which we will discuss later.

Return type—there has to be the type of value being returned from the method and data type has to be mentioned.

1. Name of method -the rules for field names apply to method names as well, but the convention is a little different.
2. parenthesis with parameter list —input parameters differentiated by comma delimiters

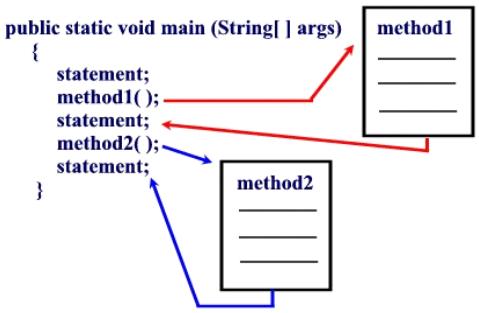


Figure 2. 2 Structure of a Function

2.3 METHOD OVERLOADING

Java can recognize the techniques with various strategy marks. for example the techniques can have same name yet with various parameters list (for example number of the parameters, request of the parameters, and information sorts of the parameters) inside a similar class.

1. We can differentiate on basis of number and type of arguments we have available.
2. We can have more than one function with same name and type and order of arguments but a compiler error will be achieved.
3. While overloading , the return type is not considered as a mean to differentiate between the functions.

If we need same kind of operations for different kind of entities we may use overloading.
We can doo overloading by different kind of methods :

1. By the methods containing the number of parameters.

```
Eg:  int add(int a, int ,b)
{
//body
}

Int add (int a, int  b, int c)
{
//body
}
```

2. Methods containing data types of parameters.

```
Eg: int add(int a, int b,int c)
{
//body
}
Double add (double a, double b, double c)
{
//body
}
```

3. Methods having different order of parameters.

```
Eg:void func(String name, int i)
{
//body
}
Void func(int i, String name)
{
//body
}
```

2.4 CONSTRUCTOR

They are needed to initialize the state of an object.
Whatever statements are inside a constructor are executed while a object is created.
They assign value to an object either it is done explicitly or the compiler does it automatically for us.
When we use new keyword to create an object , the constructor is automatically created which is the default constructor. It is used to assign values to variable and data members.
It is invoked at time of object creation.

Syntax:

```
Class Planet
{
…..
// A constructor
New Planet()
{
}
…
}
Planet obj=new Planet(); // here when object is created , default constructor is invoked.
```

Types of Constructor:

We have two types of constructor :

1. Default constructor: it has no parameters. If we dont define any constructor then the compiler makes one default constructor.
   But if we provide our own then we dont need the default constructor.
2. Parameterized Constructor: Constructor having parameters inside declarations is known as parameterized Constructor. When we want initialize our own values then this type of constructor is needed.

   There is no return value associated with a constructor since it is used for initialization.

DIFFERENCE IN CONSTRUCTOR AND METHODS.

Constructor(s) must have the same name as the class within which it is defined while it is not necessary for the method in java.

Constructor(s) do not return any type while method(s) have the return type or void if does not return any value.

Constructor is called only once at the time of Object creation while method(s) can be called any number of times.

2.5 CLASSES

Objects are individually created in a class.

Variables and methods are contained inside a class. It will have it's own . java method.

There are many access level to the class:

Table 2.1 Access Levels

Access Name	Accessibility
Private	Only in this class
Package	Only in this package
Protected	Only in this package and n all subclasses of this class
Public	Everywhere this class is available

When a certain method cannot access any of non-static blocks of class then it should be declared static. Whatever object a class contains , it inhibits the same properties and behaviours of the class. The behavoiour of the object is described by the fields and methods.

THIS KEYWORD

This is the keyword that refers to the current object.
We can use it to :

1. Refer current class instance
2. Invoke current class constructor
3. Return the current class instance
4. Used a method parameter
5. Invoke current class method
6. Argument in constructor class

2.6 DATA TYPES

Simple Datat types: short,, int, long,, char, float,, double & boolean.

Table 2. 2 Data Types

Data type	Size	Range
Long	64	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
Int	32	–2,147,483,648 to 2,147,483,647
Short	16	–32,768 to 32,767
Byte	8	–128 to 127
float	64	4.9e–324 to 1.8e+308
Double	32	32 1.4e−045 to 3.4e+038

2.7 OPERATORS

We have many types of operators in JAVA:

Arithmetic operators: arithmetic operations can be performed on data types.
Multiplication : *
Division : %
Modulo : /
Addition : +
Subtraction : -

Unary Operator :

Need only one operand in JAVA is done by the Unary Operator.
Unary minus: -
Unary Plus : +

Increment Operator :++

Post increment : value computed first then incremented

Pre increment : value incremented first then computed

Decrement Operator : --

Post decrement : value computed first then decremented

Pre decrement : value decremented first then computed

Logical not operator : ! invert boolean value

Assignment operator: it is used to assign a value to a variable.

Relational Operator : to compare values in JAVA we use relational operator .

Equal : ==
Not equal to : !=
Less than or equal to : <=
Greater than or equal to : >=
Greater than : >
Less than : <

Logical Operator:

AND-&&
OR-||
NOT- !

Ternary Operator: It is shorthand used for if-else statements.

Condition ? if true : if false

2.8 VARIABLES

- Java program has basic unit which is called variable. A variable is a combination of a type , an identifier & initializer(optional)

- they all have a defined scope,which illustrates their visibility. All variable must be declared before they are used.

SYNTAX:

```
type identifier [ = value] [, identifier [= value] ...];
```

2.9 STATIC

The word status is a keyword that can be used with class , block , method , variable. They belong to the class instead of a specific instance. TO access a static word we dont need object.

EXAMPLE:

```
Class Sample
{
Static void myFunction()
{
```

```
//statements
}
main()
{
myFunction();
}
```

Here the function myFunction can be called directly without using any object because now it has become a class level member.

STATIC VARIABLE

It is common to all objects of the class as it a class level variable. A single copy of the variable is shared by all the objects of the class created. Any changes made to the variable will be reflected in all the objects .

They are also called Class Variable.

STATIC METHODS

They can access static variables or as they are called class variable without using any instance of the class.

Both static and non static method can access static methods.

SYNTAX : static return _type method_name ();

STATIC CLASS

To make a class static , it needs to be nested. Non static members of outer class cant access static class.

2.10 STATIC VS INSTANCE METHODS

Instance Methods:
For instance methods we require and object to be created before it has to be called to execute the function definition.
We need to create an object in the class within which it is needed.

```
Syntax:
Public void func(String n)
{
//code to be executed
}
//return type can be any data type
```

-The instance methods are a property of the object and not the class .

1. Every object which we create in the class has its own copy of the instance methods.

2. They are stored on per instance basis but stored in single memory allocation.
3. They can be overridden as we have seen in run time polymorphism.

Static Method: They can be called without any creation of object since they are a property of class and not any object..
They are referenced by class name itself.

Syntax:
Public static void func(String name)
{
//code to be executed
}
//need to have static modifier
??return type can be any data type

1. The static methods are linked to class that is they exist without any need to create an object.
2. They are share by all objects of the class.
3. Any changes made will be reflected in all the classes.
4. They cannot be overridden and hence follow static binding.

2.11 WRAPPER CLASSES

We have mentioned various wrapper class of primitive data types below. Wrapper classes convert primitive data types into objects.

Primitive type	Wrapper class
boolean	Boolean
byte	Byte
char	Character
float	Float
int	Integer
long	Long
short	Short
double	Double

Figure 2. 3 Wrapper Class Conversion

They are needed in API collections. We should use wrapper classes when we need object is instead of primitive data types.

AUTOBOXING

Java compiler makes automatic conversion between primitive data types and their respective wrapper classes. This is autoboxing.

If the conversion is vice versa then it is unboxing.

EXAMPLE: Character char='z';

Compiler performs autoboxing when a primitive value is :

1. Parameter to some method which instead expects a object of wrapper class.
2. Compiler performs unboxing when object of wrapper class s :
3. Passed as parameter to a function which is expects a value of primitive type of corresponding type.

2.12 FLOW CONTROL STATEMENTS

IF-THEN : It is the simplest control flow statement. It executes statements depending on which condition is correct.

IF-ELSE: It provides and alternate executing statement if conditions come out to be false. if-else-if-else: to make a decision tree sort of statements and executions.

SWITCH : It gives option of range of values for our variable .

WHILE : It is a loop statement to execute a set of statements till a certain condition is true . As soon as the condition is false , the control comes out of the loop. We need a increment variable to keep a check on a condition is true.

```
while (true){
    // some java statements
}
```

Figure 2. 4 Syntax of WHILE statement

FOR LOOP : It repeats a set of JAVA statements. It runs a set of code as long as a certain condition is true.

```
for (initialization; termination condition; increment/decrement) {
    //java statement(s)
}
```

Figure 2. 5 Syntax of FOR loop

BREAK : It isused to terminate from a loop .

CONTINUE : It is used to skip the current execution of a statement inside a loop and move further .

```
while (someCondition) {
    if (someOtherCondition) {
        continue;
    }
    // Do something
}
```

Figure 2. 6 CONTINUE STATEMENT

2.13 ACCESS MODIFIERS

It is used to limit the access of a class , constructor , data variable or a method in another class .

We have four types of modifiers :

1. Default:
2. Private:
3. Protected:
4. Public:

The details of access are given in this table :

	Class	Package	Subclass (same package)	Subclass (diff package)	Outside Class
public	Yes	Yes	Yes	Yes	Yes
protected	Yes	Yes	Yes	Yes	No
default	Yes	Yes	Yes	No	No
private	Yes	No	No	No	No

Figure 2. 7 Details of Access Modifiers

2.14 PACKAGES

It is a group of interfaces, classes and other packages as well.

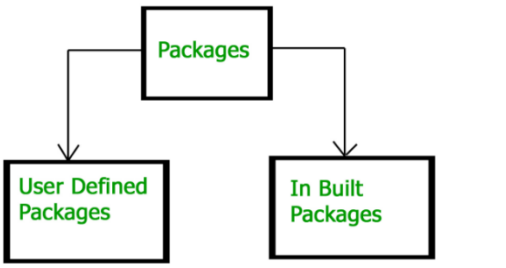


Figure 2. 8 Types of Packges

SYNTAX: import java.util.Scanner

In this syntax : the top package is java , the subpackage is util, Scanner is a class inside util package .

Why do we use packages :

- Name ambiguity: to avoid collision of same name class ,we can use packages
- Reusabilty: Instead of writing a certain code repeatedly , we can use packages to import them inside any class.

EXAMPLE:

Java Packages: java.io.*, java.lang.*

User defined : we can create packages with any name. It should be the first statement inside your program.

2.15 CONCEPTS OF OOP

2.15.1 INHERITANCE

When one class s acquired by another class, then it is defined as inheritance. It is generally to provide reusability inside a class. The functionalities can be incorporated from one class to another class.

Parent Class

It is the class from which some other class inherits functions and properties.

Derived or Child class

The class which inherits some features from parent class.

The code which is already written inside parent class need not be repeated inside child class.

Syntax:

```
class XYZ extends ABC
{
}
```

Figure 2. 9 Example of Inheritance

TYPES OF INHERITANCE :

1. **Single inheritance** : A derived class extends another parent class.

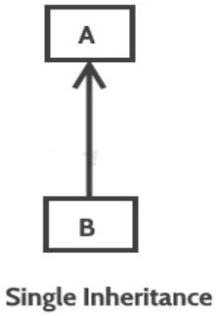


Figure 2. 10 Single Inheritance

2. **Multilevel Inheritance :** A derived class extends parent class which in turn extends another parent class.

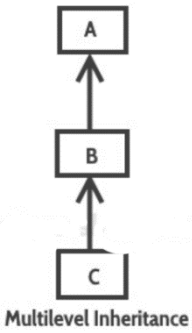


Figure 2. 11 Multiple Inheritance

3. **Hierarchical Inheritance:** Where more than one class extends another class.

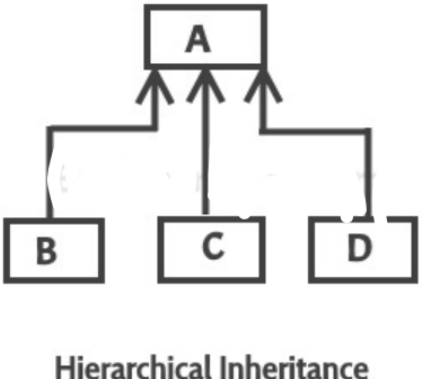


Figure 2. 12 Hierarchical Inheritance

4. **Mulitple Inheritance :** When onederived class is inherited by more than one parent class. Java doesnt support multiple inheritance .

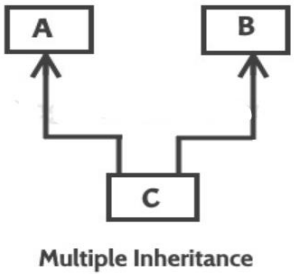


Figure 2. 13 Multiple inheritance

2.15.2 ABSTRACTION

When the details are hidden and functionalities are shown to the user , it is defined as abstraction.

It is used to focus only on the object and not how the work is done in background.

ABSTRACT CLASS

When we declare a class as abstract , it is abstract class. A abstract class should be extended and it cannot be instantiated.

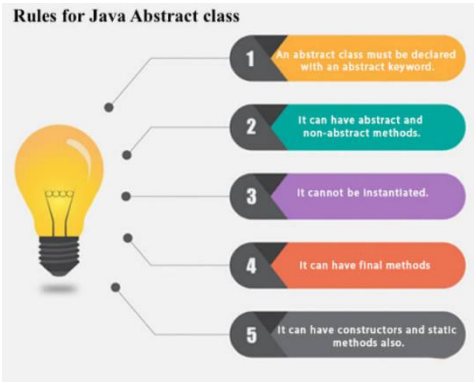


Figure 2. 14 RULES FOR ABSTRACT CLASS

ABSTRACT METHOD

When a method does not contain any implementation and is declared with the keyword abstract then it is known as abstract method.

SYNTAX: abstract void abfunstion(); // no method body

An abstract class can have abstract method , method body (when it is non abstract method0 , constructor , even main () method.

2.15.3 POLYMORPHISM

This is a feature of OOPs that gives us ability to perform a single action in various different ways. Suppose we want to calculate area of different figures in Area class with calcarea() method. The shapes could be circle, rectangle , square , parallelogram etc . Suppose we have one Area class and all other shapes as it's

subclasses. The methods to calculate area could have different implementation in different subclasses with differernt names.

EXAMPLE:

```
Public class Area
{
public void calcArea(){
System.out.println("Area can be calculated in base class");
}
}

Public class Area extends Circle{
@override
Public void calcArea(){
System.out.println("Area can be calculated of the circle");
}
}
```

We have two types polymorphism in Java:

1. Method Overloading : It is compile-time or static polymorphism.

2. Method overriding: It is runtime or dynamic polymorphism.

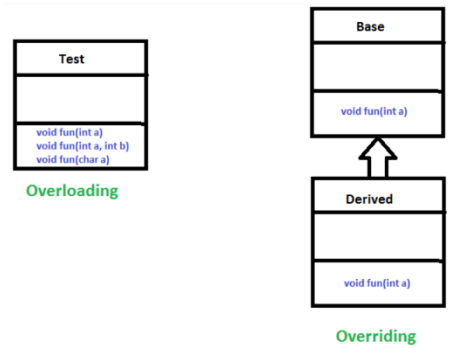


Figure 2. 15 Method Overloading and Method Overriding

When there are many functions with same name but they differ in number of parameters or the data type of arguments , then it is said to be overloading.

When the derived class has a separate implementation of function of base class , then the base function is said to be overridden.

2.15.4 ENCAPSULATION

It is simply states as the binding of object state which is called fields and the behaviour or methods together.

When we create any class , we are implementing encapsulation.

Encapsulation is performed to hide from users the details of implementation.

How to implement Encapsulation ?

1. We have to make the variables private so they cannot be accessed from outside class directly .
2. We have to use getter and setter to get and set the values of fields .
   EXAMPLE:

   ```
   Public int get()
   {
   Return value;
   }

   Public int set()
   {
   variable=value;
   }
   ```

ADVANTAGES OF ENCAPSULATION:

1. The maintainability and flexibility and reusability is enhanced through encapsulation.
2. If we dont define setter methods then field can be made read only.
3. If getter methods are not defined then the field can only be set write only .So it is not possible for outside classes to modify values.
4. The user does not need to know behind the scene implementation. Encapsulation is also defined as Data Hiding.

2.16 MULTITHREADING

To understand multithreading we need to know what thread means. A thread is the tiniest poriton of a process that can be run simultanoeusly with the other thread of the same process. They have independent execution.

common memory is shared by threads . To execute multiple threads concurrently is called Multithreading.

USAGE OF MULTITHREADING:

1. Two or more parts of a programs can be run parallely and it increases the efficiency of a CPU
2. Thread that are running concurrently or simultaneously utilize the maximum CPU time .
3. A thread can have many states.
   The life cycle of a thread s depicted in following diagram:

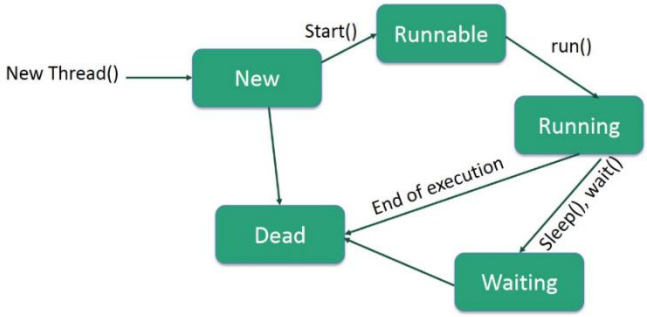


Figure 2. 16 Multithreading

2.17 STRING HANDLING

String is defined as sequence of characters. It is immutable object which means it cannot be modified,once it has been made. We can create string using:

1. String literal
2. new keyword
   EXAMPLE:

   String string1=”Hello”;

   String string2= new String(“World”);

VARIOUS STRINGS METHODS:

1. char charAt(int index): returns character at specified index.
2. boolean equals(Object obj): returns true if string equal to another string else false.
3. compareTo(String string) : compares two strings using uni code equivalent of the characters.
4. indexOf(int ch):return index in the string of a specific character.
5. substring(int beginIndex): returns the substring of a string.
6. substring(int beginIndex, nt lastIndex): return substring between specified indexes.
7. concat(String str) : contacts specified string to original string.
8. replace(char OldChar,char newCHar): replaces old character with new character.
9. toUpperCase(): converts string to upper case.
10. to LowerCase(): converts string to lower case.
11. char[] toCharsArray(): converts the string to a character array .
12. valueOf(): return string representation of arguments that are passed like float, long, double , int etc.
    String trim(): return string after omitting the preceding and following spaces

2.18 ARRAYS

They are objects which can sore multiple values in contiguous memory locations . It can hold primitive data types as well as object references.

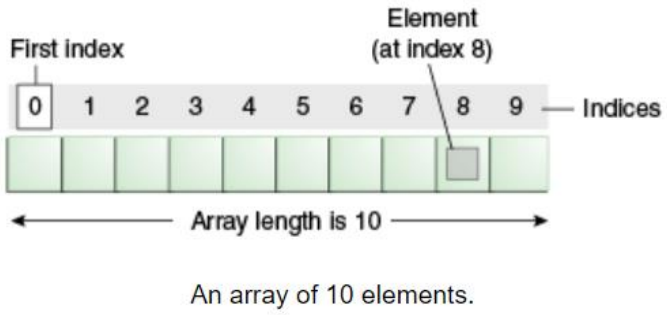


Figure 2. 17 Structure of Array

FEATURES OF ARRAYS:

1. They are objects.
2. reference variables of other objects can be stored in array.
3. They created on the heap and are dynamic.
4. The length of an array is fixed.

Different ways to declare an array:

```
Int firstArray[];
        Or
Int [] secondArray;
        Or
Int thirdArray[]={1,2,3,4,5};
```

PASSING ARRAY TO METHOD :

We can pass array to methods

```
EXAMPLE:
Class ArraysClass
{
main()
{
Int arr[];
sum(arr);
}
Void sum(int[] arr)
```

{

Body of function

}

}

2.19 EXCEPTION HANDLING N JAVA

It allows us to hande runtime errors caused by exceptions. An exception is an event which is unwanted and it disrupts the normal flow of a program. The program gets terminated as soon as an exception occurs.

With the use of exception , it gets easier to inform the user about the reason behind the interrupted flow of a program. An exception can occur due to network connection problem , access to invalid indexes, etc.

TYPES OF EXCEPTION:

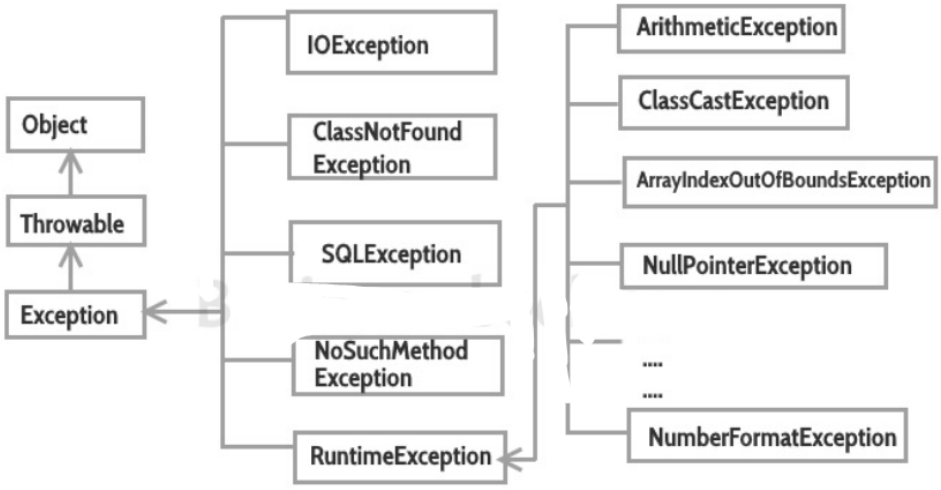


Figure 2. 18 Types of Exception

To perform exception handling we can use try catch :

TRY: Try is used to cover those statements where an exception can occur.

It is always followed by a catch block which handles those exceptions.

SYNTAX:

Try

{

//statements that mau cause exception

}

CATCH: It is where the exceptions occurring in try block are resolved and handled. We can catch different type of exceptions in different catch block .

SYNTAX:

Try

{

//statements that mau cause exception

}

Catch (exception(type) e(object))

{

// erro handling code }

2.20 THROW and THROWS keyword

1. When we want to explicitly throw an exception in a method or any block inside the code then we use the throw keyword.
2. It is either a checked or an unchecked exception.
3. It is primarily used to throw custom exceptions.

Eg:
Throw new NullPointerException(“”);
By this we must have instance by throwable type or a subclass of throwable.
User defined exceptions typically extend Exception CLass.
When compiler encounters a thro keyword it stood immediately and finds the try block and check if it has some catch block implementation.

Eg:
Void func()
{
Try
{
Throw new ArithmeticWException(“demo”);
}
catch(NullPointerException e)
{
//body of catch
}

Tghrows keyword suggests that methods might throw any of the listed exceptions.
The exceptions to these methods have to handle using try - catch block.
Syntax:
Type method_name(parameter) throws exception_list
It is a comma separated list.
Eg:
Static void main(String args[]) throws InterruptedException
{

```
Thread.sleep(10000);
….
}
```

- When we have checked exceptions then only we can use throws keyword while usage for unchecked is meaningless through throws.
- It is required only to convince the compiler as it does not prevent any abnormal termination of the program.
- Information to the caller can be provided by the help of throws keyword.

2.21 GARBAGE COLLECTION IN JAVA

In C++ the responsibility for both creation and destroying objects is in the hands of the programmer but in JAVA , we need not bother about it at all.
Because for all those objects which are no longer used then the garbage collector destroyed those objects. The main objective of Garbage Collector would be to free any heap memory by destroying the unreachable objects.

When we have an object which does not contain any reference to it then it is called an unreachable object.
Eg:

```
FLoat f=new Float(4.0);
// this object f is reachable via reference in f
f=null;
//now the object is no longer reachable.
```

The object is eligible for garbage collection only if it is unreachable.

2.22 FILE HANDLING

The most affordable and popular programming language JAVA provides extensive support in various data base functionalities as well as sockets etc. File Handling is necessary to perform different tasks on file like read , write , etc .
It implies that we can read and write into data. Java.io.package allows us to maiupulate different files at different times. We need object of the class to specify the filename or directory name.
Eg:

```
Import java.io.File
File file_obj=new File(“file_name.txt”);
```

Here we use the phenomenon of I/O operations on file .
Streams:
It is sequence of data which can be in two forms :
Byte Stream:
It comprises of byte data. When we have input provided and it can be execute d with byte data then that is called file handling with byte stream.

Character Stream :
It contains character. When we process data with chartered as input then it is called file handling with characters stream.

We have some of the JAVA file methods :

Method	Type	Description
canRead()	Boolean	It tests whether the file is readable or not
canWrite()	Boolean	It tests whether the file is writable or not
createNewFile()	Boolean	This method creates an empty file
delete()	Boolean	Deletes a file
exists()	Boolean	It tests whether the file exists
getName()	String	Returns the name of the file
getAbsolutePath()	String	Returns the absolute pathname of the file
length()	Long	Returns the size of the file in bytes
list()	String[]	Returns an array of the files in the directory

Figure 2. 19 File Handling Methods

File Operations :
We can have 4 types of operations on a file.
1.Create a File

To create a file we can use createNewFiles() method. It returns true if successfully file has been created and false if file is already in the system.

2. Get File Information:

We have certain methods like getName(), canWrite() etc to achieve information regarding the file.

3. Write to a File:

We have syntax like :
FileWriter myFile =new FileWriter(" C: FileHandlingMyFile.txt");

4.Read from a File:

Here we can use Scanner class to read input from console .
We can create Scanner class object with syntax: Scanner obj=new Scanner(System.in);

CHAPTER 3 PROGRAMMING FUNDAMENTALS USING PYTHON

3.1 INTRODUCTION

Python s for general purpose translation, interoperability, object orientation, and advanced programming language. It was developed by Guido van Rossum in 1985- 1990. Like Perl, Python's source code is also available under the GNU General Public License (GPL). This tutorial provides enough insight into the programming language of Python.

Why to Learn Python?

Python s is a high quality translated, interactive and object-oriented language. Python is designed to be very readable. It uses English keywords more often there as some languages use punctuation, and has fewer syntactic properties than other languages.

the key advantages of learning Python:

- Interpreted language – An interpreter processes the content.No compilation is needed.
- Interactive language− We can interact directly with the prompt of Pythion.
- Object-Oriented − The technique of language is much similar to OOPs
- Beginner's Language − These language is helpful for beginners.

Characteristics of Python

Following are important characteristics of Python Programming −

- Functional & structured programming types are supported in Python.
- Usage can be transformed to scripting type of language as well as byte code .
- Dynamic type of rechecking is allowed in this language..
- Collection of garbage is automatic.

Applications of Python

- Few keywords & a defined syntax makes it a easy language.
- The structure of code is clearly defined hence readability is very clear.
- The amount of maintenance required is quite less.
- It has a huge amount of portable library.
- It allows testing and debugging features
- It has portable hardware platform.
- The audience can customize their tools to develop a program in Python.
- All commercial database are provided interfaces.

3.2 WRITIING A PYTHON SCRIPT

There are two main windows that we will work with n DLE: the interactive window, which is the one that opens when we start DLE, and the script window. We can type code into both the interactive and script windows. The difference between the two is how the code is executed.

THE INTERACTIVE WINDOW

The interactive window contains a Python shell, which is a textual user interface used to interact with the Python language. Hence the name "interactive window." When you first open DLE, the text displayed looks something like this:

```
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24)
[MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 3. 1 Interactive Window

The first line tells you what version of Python is running. In this case, DLE is running Python 3.8.1. The second and third lines give some information about the operating system and some commands you can use to get more information about Python. The >>> symbol in the last line is called the prompt. This is where you will type n your code. Go ahead and type 1 + 1 at the prompt and press Enter. When you hit Enter , Python evaluates the expression, displays the result 2, and then prompts you for more input:

```
>>> 1 + 1
2
>>>
```

Figure 3. 2 Output

THE SCRIPT WINDOW

Scripts are written using IDLE's script window. You can open the script window by selecting File New File from the menu at the top of the interactive window.

Any output generated by code run in the script window is displayed in the interactive window, so you may want to rearrange the two windows so that you can see both of them at the same time. in the script window, type in the same code you used to print "Hello, world" in the interactive window. Just like the interactive window, code typed into the script window is highlighted.

Once the script is saved, all you have to do to run the program is select Run Run Module from the script window and you'll see Hello, world appear in the interactive window.

3.3 VARIABLES

There are certain rules or naming conventions for naming variables. The following are the rules:

1. Reserved key words such as if, else, and so on cannot be used for naming
2. Variable names can begin with _, $, or a letter
3. Variable names can be in lower case and uppercase
4. Variable names cannot start with a number White space characters are not allowed in the naming of a variable.

You can assign values to the variable using = or assignment operator.

3.4 DATA TYPES:

There are many types of data, such as numbers, strings, character, images, and so on.
Data types can be broadly categorized into five different types, listed as follows:
Numbers
String
Tuples
List
Dictionary

NUMERIC DATA TYPE: Numeric data types or numbers . There are commonly four numeric information types in Python. They are whole numbers, long numbers, drifting point numbers, and complex numbers. Whole numbers and long whole numbers Integers incorporate zero, the entirety of the positive entire numbers, and the entirety of the negative entire numbers. The translator first checks the articulation on the correct hand side of the task administrator and afterward ties the incentive with its variable name; this procedure is named as factor definition or introduction. The int or number information type ranges from - 231 to (231-1); the main short sign shows the negative qualities.

FLOATING POINT DATA TYPE: Numbers with specific places after the decimal point are alluded to as floating point numbers in the programming language:
The floating point number sort goes around from - 10 to 10^308 and has 16 digits of exactness.
There are two different ways to compose a floating point number. It very well may be composed utilizing standard decimal documentation or logical documentation. Logical documentation is regularly valuable for referencing extremely enormous numbers.

3.5 OPERATORS IN PYTHON

Python supports the following types of operators:

1. Arithmetic operators.
2. Comparison operators
3. Assignment operators
4. Bitwise operators
5. Logical operators
6. Membership operators
7. Identity operator

1. ARITHMETIC OPERATORS

It comprise operands and operators

Operator	Description
**	**Exponent**: Performs exponential (power) calculations on operands
*	**Multiplication**: Performs multiplication between operands
/	**Division**: Performs division between operands
%	**Modulus**: Performs modulus division between operands
+	**Addition**: Performs addition between operands
-	**Subtraction**: Performs subtraction between operands

Figure 3. 3 Arithmetic Operators

2. COMPARISON OPERATOR

It returns true or false

Operator	Description
==	Checks the equality
<	Returns `True` if the left-hand side operand is less than the right-hand side operand
>	Returns `True` if the left-hand side operand is greater than the right-hand side operand
<=	Returns `True` if the left-hand side operand is less than or equal to the right-hand side operand
>=	Returns `True` if the left-hand side operand is greater than or equal to the right-hand side operand
!=	Returns `True` if the left-hand side operand is not equal to the right-hand side operand
<>	Returns `True` if the left-hand side operand is not equal to the right-hand side operand

Figure 3. 4 Comparison Operator

3. ASSIGNMENT OPERATOR

It is not only used to assign values to variables but are often used in combination with arithematic operators

+=	`x+=y` is equivalent to `x=x+y`
-=	`x-=y` is equivalent to `x=x-y`
*=	`x*=y` is equivalent to `x=x*y`
/=	`x/=y` is equivalent to `x=x/y`
**=	`x**=y` is equivalent to `x=x**y`

Figure 3. 5 Assignment Operator

4. BITWISE OPERATOR

Python supports bitwise operation. Basic bitwise operators are:

Operator	Description
|	Performs binary *OR* operation
&	Performs binary *AND* operation
~	Performs binary *XOR* operation

Figure 3. 6 Bitwise Operators 1

There are some more bitwise operators

^	Performs binary one's complement operation
<<	**Left shift operator**: The left-hand side operand bit is moved left by the number specified on the right-hand side
>>	**Right shift operator**: The left-hand side operand bit is moved right by the number specified on the right-hand side

Figure 3. 7 Bitwise Operator 2

5. LOGICAL OPERATORS

It supports logical operators like AND, OR , and NOT:

Operator	Description
`and`	Returns `True` if both the right-hand and left-hand sides of the operator are true
`or`	Returns `True` if any side, either the right-hand side or the left-hand side, of the operator is true
`not`	If condition in the `not` operator is `True`, the `not` operator makes it `False`

Figure 3. 8 Logical Operator

6. MEMBERSHIP OPERATORS

Python has two membership operators to test the membership in a sequence, such as a string, list, tuple, and others:

Operator	Description
`in`	Returns `True` if the specified operand is found in the sequence
`not in`	Returns `True` if the specified operand is not found in the sequence

Figure 3. 9 Membership Operator

7. IDENTITY OPERATOR

Given in the table are the two identity operators:

Operator	Description
`is`	Returns `True` if two variables point to the same object and `False`, otherwise
`is not`	Returns `False` if two variables point to the same object and `True`, otherwise

Figure 3. 10 Identity Operator

3.6 OPERATOR PRECEDENCE

Operators with the highest precedence are placed on the top:

Table 3. 1 Order Precedence

Operator	Description
`( )`	Parentheses
`x[index],x[index1:index2],f(arg...),x.attribute`	Subscription, slicing, call, and attribute reference
`**`	Exponentiation
`+x, -x, ~x`	Positive, negative, and bitwise *NOT*
`*, /, %`	Multiplication, division, and remainder
`+, -`	Addition and subtraction
`<<, >>`	Shifts
`&`	Bitwise *AND*
`^`	Bitwise *XOR*
`|`	Bitwise *OR*
`in, not in, is, is not, <, <=, >, >=, !=, ==`	Comparisons, including membership tests and identity tests
`not x`	Boolean *NOT*
`and`	Boolean *AND*
`or`	Boolean *OR*
`if...else`	Conditional expression
`lambda`	Lambda expression

Operators that have the same precedence are evaluated from left to right, except for comparisons and exponentiation.

3.7 STRINGS AND STRING METHODS

Collections of text in Python are called strings. Special functions called string methods are used to manipulate strings. There are string methods for changing a string from lowercase to uppercase, removing whitespace from the beginning or end of a string, or replacing parts of a string with different text, and many more.

THE STRING DATA TYPE

Strings are one of the fundamental Python data types. The term data type refers to what kind of data a value represents. Strings are used to represent text. Strings are a fundamental data type because they can't be broken down into smaller values of a different type. Not all data types are fundamental. Strings have three properties that you'll explore in the coming sections:

1. Strings contains characters, which are individual letters or symbols.
2. Strings have a length, which is the number of characters contained in the string.
3. Characters in a string appear in a sequence, meaning each character has a numbered position in the string.
Whenever you create a string by surrounding text with quotation marks, the string is called a string literal. The quotes surrounding a string are called delimiters because they tell Python where a string begins and where it ends.

```
string1 = 'Hello, world'
string2 = "1234"
```

Figure 3. 11 String Data Type

Determine the Length of a String

The number of characters contained in a string, including spaces, is called the length of the string. For example, the string "abc" has a length of 3, and the string "Don't Panic" has a length of 11. To determine a string's length, you use Python's built-in len() function. To see how t works, type the following into IDLE's interactive window:

```
>>> len("abc")
3
```

Figure 3. 12 Length function

STRING CONCATENATION

Two strings can be combined, or concatenated, using the + operator:

```
>>> string1 = "abra"
>>> string2 = "cadabra"
>>> magic_string = string1 + string2
>>> magic_string
'abracadabra'
```

Figure 3. 13 String Concatenation

In this example, string concatenation occurs on the third line. string1 and string2 are concatenated using + and the result is assigned to the variable magic_string.

STRING INDEXING

Each character in a string has a numbered position called an index. You can access the character at the Nth position by putting the number N in between two square brackets ([and]) immediately after the string:

```
>>> flavor = "apple pie"
>>> flavor[1]
'p'
```

Figure 3. 14 String Indexing

flavor[1] returns the character at position 1 in "apple pie", which is p.
The following figure shows the index for each character of the string "apple pie":

	a	p	p	l	e		p	i	e
	0	1	2	3	4	5	6	7	8

Figure 3. 15 Index of each character

Strings also support negative indices:
The last character in a string has index -1, which for "apple pie" is the letter e. The second-to-last character has index -2, and so on.
The following figure shows the negative index for each character in the string "apple pie":

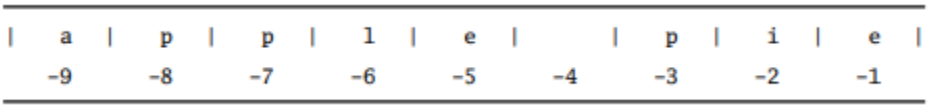


Figure 3. 16 Negative index of each character

STRING SLICING

You can extract a portion of a string, called a substring, by inserting a colon between two index numbers inside of square brackets, like this:

```
>>> flavor = "apple pie"
>>> flavor[0:3]
'app'
```

Figure 3. 17 String Slicing

flavor[0:3] returns the first three characters of the string assigned to flavor, starting with the character with index 0 and going up to, but not including, the character with index 3. The [0:3] part of flavor[0:3] is called a slice. String slices can be confusing because the substring returned by the slice includes the character whose index is the first number, but doesn't include the character whose index is the second number.

3.8 PYTHON COLLECTIONS

We have four type of collections:

LIST

List makes it possible to make change and order in a collection.

```
EXAMPLE:    mylist = ["cloud", "sky", "stars"]
            print(mylist)
```

TUPLE

Tuples are same as list but the are written in round brackets and access method is different

```
EXAMPLE: mytuple = ("cloud", "star", "sky")
         print(mytuple)
```

SET

Collection which is unordered and unidexed.

```
EXAMPLE: myset = {"cloud", "sky", "star"}
         print(myset)
```

DICTIONARIES

Collection which is not ordered , indexed and can be changed .They are mentioned within curly braces and they come in key value pairs.

```
EXAMPLE:      mydict = {
 "day": "13",
 "month": "May",
 "year": 2020
}
print(mydict)
```

3.9 PYTHON LOOPS:

Python supports different conditions which can be logical:

- Equals: a=b
- Not Equals: a!=b
- Less than:a<b
- Less than or equal to: a<=b
- Greater than:a>b
- Greater than or equal to: a>=b

IF STATEMENT

We use following structure

```
EXAMPLE:value1=29
        value2=89
         if value2>value1:
            print("value2  is greater than value1")
```

In this example 2 values are compared and the greater one is printed.
To define a scope inside a python program it depends upon the indentation. We dont use curly brackets much like other programming languages.

ELIF STATEMENT

```
EXAMPLE:value1 =1 33
        value2 =2 33
         if value2 > value1:
           print("value2  greater than value1")
         elif value1 == value2:
            print("value1 and value2  equal")
```

ELSE STATEMENT

The else keyword catches anything which isn't caught by the preceding conditions.

```
EXAMPLE:
aa = 1200
bb = 133
if bb > aa:
  print("bb greater than aa")
elif aa == bb:
  print("aa & bb are equal")
```

```
else:
  print("aa  greater than bb")
```

We have two loop commands which are primitive:

- While
- For

WHILE LOOP

When we want to execute block of statements tilla certain condition is right.

EXAMPLE:

Print d as long as d is ggreater than 7

```
d = 18
while d >7:
print(d)
d -= 1
```

BREAK STATEMENT

To come out of a loop body

EXAMPLE:

Exit the loop when d=7

```
d= 1
while d < 10:
  print(d)
  if d == 7:
    break
  d += 1
```

CONTINUE STATEMENT

When we dont want to execute current iteration and jump onto next iteration

Eg:

Continue to next line when d=7

```
d =1
while d < 10:
  d+= 1
  if d == 7:
    continue
  print(d)
```

FOR LOOP

To iterate over a set of tuples, list or dictionary or characters within a string we canuse for loop

EXAMPLE:

```
brands=["Timex", "Sonata", "G-Shock"]
for i in brands:
  print(i)
```

RANGE() FUNCTION

When we want the loop to run only a specified number of times we use range function.

The increment is 1 by default.

Eg:

Using the range() function

```
for i in brands(6):
    print(i)
```

3.10 PYTHON FUNCTIONS

When we want a block of code to be executed many times whenever required we use the feature of function. Parameters or arguments can be passed into function.

CREATING A FUNCTION

```
EXAMPLE:def myfirst_function():
            print("My first function")
```

CALLING A FUNCTION

```
EXAMPLE:    def myfirst_function():
              print("My first function")

            myfirst _function()
```

ARGUMENTS

Details can be transferred into functions as arguments.

Issues are defined after the function name, within parentheses. You can add as many arguments as you want, just split by comma.

The following example has a function for a single argument (firstname). When a function is called, we pass the first name, used inside the function to print the full name:

```
EXAMPLE:    def myfirst _function(firstname):
              print(firstname + " Refsnes")

            my_function("Emil")
            my_function("Tobias")
            my_function("Linus")
```

RETURN VALUES

it is used to return some value from a function:

EXAMPLE:
```
def mysecond_function(x):
    return 2* x

print(mysecond_function(6))
print(mysecond_function(78))
print(mysecond_function(23))
```

RECURSION

Python also accepts workflow, which means a defined function can be expensive. Going back is a general concept of mathematics and programming. It means the job is expensive. This has the advantage of the definition that you can enter using data to get the result. The developer should be very careful about duplication because it can be very easy to write work that has never been stopped, or that uses excess memory or processor power. However, if properly written, repetition can be an effective and great way to organize.

3.11 PYTHON CLASSES/ OBJECTS

Python is operated with help of objects which have their own methods and variables

CREATING A CLASS

EXAMPLE: CLass name-MyFirstClass, property- x1:

```
class MyFirstClass:
  x1 = 5
```

CREATING OBJECT

MyFirstClass class is used to create objects:

EXAMPLE: object -obj1, print the value of x:

```
obj1 = MyFirstClass()
```

OBJECT METHODS

Methods are contained inside an object.

To create method in the Student class:

EXAMPLE:

```
class Student:
  def __init__(obj, first_name, stdid):
    obj.first_name = name
    obj.stdid = id

def my_func(obj):
   print(" my name is " + obj.name)

p1 = Student("Mohit", 161007)
p1.my_func()
```

The obj parameter is current object of class and is used to pass the variables .

THE SELF PARAMETER

To access a variable that belongs to a class we need self parameter.
It can be named anything but the only restriction is that it has to be the first parameter inside the function.
EXAMPLE:

```
class Student:
  def __init__(myobject, name, age):
    myobject.name = name
    myobject.age = age

  def my_func(abcd):
    print(" my name is " + abcd.name)

p1 = Student("Mohit", 22)
p1.my_func()
```

MODIFY OBJECT PROPERTIES

EXAMPLE : Set the name of p1 to Varun:
p1.name= Varun

DELETE OBJECT PROPERTIES

delete prop of objects:
EXAMPLE: Delete the age

```
del p1.age
```

DELETE OBJECTS

deleting objects by del keyword:
EXAMPLE:
Deleting p1 object:

```
del p1
```

THE PASS STATEMENT

We cannot have empty definitions of class, but to avoid getting any type of error , you can use pass statement.

EXAMPLE:

```
class Student:
  pass
```

3.12 PYTHON INHERITANCE

Inheritance provides us the basic feature of inheriting some features which can be used in a class from any existing class , which not only provides reusability but also saves time in recalling many methods.It also provides using features in varied way with varied implementations.

CREATING A PARENT CLASS

We can declare parent class like any other class

EXAMPLE:

```
class Student:
  def __init__(obj, firstname, lastname):
    obj.firstname = firstname
    self.lastname = lastname

  def print_name(obj):
    print(obj.firstname,obj.lastname)

x =Student("Mohit", "Prateek")
x.print_name()
```

CREATING CHILD CLASS

Heree derived class inherits features from parent class

EXAMPLE:

Create a class named Players, inheriting the properties Student class:

```
class Players(Person):
  pass
```

Here we don't want to add any properties to class so we used pass statement

Now the Player class and Student properties and methods are same.

EXAMPLE:

```
 Use the Players classamd create object:
x = Players("Mrunal", "Puneet")
x.print_name()
```

USE THE SUPER() FUNCTION

Using super() method we can automatically let derived class inherit all features and methods of parent class.

EXAMPLE:

```
class Players(Student):
  def __init__(obj, firstname, lastname):
    super().__init__(firstname, lastname)
```

3.13 SOFTWARE USED

SPYDER

Spyder is a powerful science environment written in Python, Python, and is also designed for scientists, engineers and data analysts. offers a unique combination of advanced planning, analyzing, debugging, and profiling software for a comprehensive tool for development with data analysis, collaboration, in-depth testing, and the best visualization capabilities of the scientific package.

In addition to its many built-in features, its capabilities can be further expanded through its plugin and API. In addition, Spyder can also function as a PyQt5 library, allowing developers to build on its functionality and interactive features such as the interactive console.

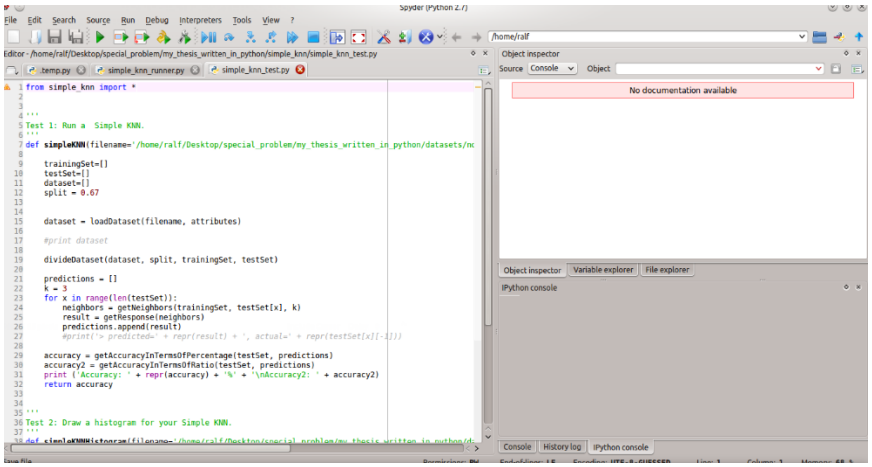


Figure 3. 18 Spyder DE

CONCLUSION

With the help of various types of commands in sql like DDL, DML, DCL, TCL etc. we were able to develop a comprehensive understanding of database creation and manipulation. In the midst of handling databases in clouds we need to handle a large amount of data and given the basic knowledge provided by this internship we will be able to further enhance cloud concepts.

The JAVA concepts expanding object oriented programming helped us in grasping the inticracies of inheritance , encapsulation , string handling and many other conceptw which not only strengthen our coding but also will help us in developing java based web apps in future.

The study of Python concepts which is becoming more and more relevant in cloud platforms has made it easier to analyse and comprehend useful information in data.This basic knowledge of python has helped to perform operations which can help in manipulation and identification of information from data.Together with SQL, JAVA and Python we were able to strengthen our basic concepts which will form the basis for cloud computing technologies in AWS.

PUBLICATION

1. Shruti Jain, Manasvi Kashyap, Mohit Garg, Shailu Srivastava, "Design and Simulation of Optimum FIR Digital Filters using Windowing Technique", The 3rd International Conference on Recent Innovations in Computing (ICRIC-2020), 20th -21st March, 2020 , Central University of Jammu, J & K **(Accepted).**

REFERENCES

1. https://www.w3schools.com/sql/sql_where.asp
2. https://www.tutorialspoint.com/sql/index.htm
3. https://www.javatpoint.com/dbms-tutorial
4. https://beginnersbook.com/2015/04/dbms-tutorial/
5. https://www.tutorialspoint.com/java/index.htm
6. https://www.w3schools.com/java/
7. https://www.javatpoint.com/java-tutorial
8. https://www.mysql.com/products/workbench/
9. https://www.python.org/
10. https://www.w3schools.com/python/
11. https://www.tutorialspoint.com/python/index.htm
12. https://www.learnpython.org/
13. https://www.spyder-ide.org/

PLAGRISM REPORT

ORIGINALITY REPORT

20%	13%	3%	16%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.geeksforgeeks.org Internet Source	2%
2	Submitted to University of Maryland, University College Student Paper	1%
3	www.melbpc.org.au Internet Source	1%
4	www.w3schools.com Internet Source	1%
5	gvpcew.ac.in Internet Source	1%
6	Submitted to Engineers Australia Student Paper	1%
7	Submitted to Jaypee University of Information Technology Student Paper	1%
8	Submitted to International School of Management and Technology (ISMT), Nepal Student Paper	1%

PROJECT REPORT UNDERTAKING

I Mr. /Ms. **MOHIT GARG** Roll No **161007** Branch **ELECTRONICS AND COMMUNICATION TECHNOLOGY** is doing my internship with **COGNIZANT TECHNOLOGY SOLUTIONS** from **7th FEBRUARY 2020** to **5th JUNE 2020**.

As per procedure I have to submit my project report to the university related to my work that I have done during this internship.

I have compiled my project report. But due to COVID-19 situation my project mentor in the company is not able to sign my project report.

So I hereby declare that the project report is fully designed/developed by me and no part of the work is borrowed or purchased from any agency. And I'll produce a certificate/document of my internship completion with the company to TnP Cell whenever COVID-19 situation gets normal.

Signature mohit Garg 30/05/2020

Name MOHIT GARG

Date 30th MAY 2020

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 14/07/2020

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | **B.Tech Project Report** | Paper

Name: Mohit Garg **Department:** ECE **Enrolment No:** 161007

Contact No. 8418955863 **E-mail:** mohit201097@gmail.com

Name of the Supervisor: Dr. Shruti Jain, JUIT Solan
MA Manoprincy Y, Cognizant Technology Solutions

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): SQL AND PYTHON : TOOLS IN CLOUD COMPUTING

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

- Total No. of Pages = 74
- Total No. of Preliminary pages = 11
- Total No. of pages accommodate bibliography/references = 3

mohit garg

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at **20** % (with 5 words). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) 15/7/2020

Signature of HOD 28/07/2020

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Abstract & Chapters Details	
	• All Preliminary Pages • Bibliography/ Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Page counts	
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete Thesis/Report in (PDF) & DOC (Word File) through your Supervisor/Guide at plagcheck.juit@gmail.com