TERA DATA

NOTES

BY

DATAWARE HOUSING TECHNOLOGIES

5/12/10

TERADATA 13 VIEWPOINT

OPPORTUNITIES:-

1. TERADATA DEVELOPER
2. TERADATA DBA
3. TERADATA TESTER [ETL TESTING]
4. DWH[ETL (OR) REPORTING] +TERADATA
5. .NET/JAVA + TERADATA
6. SSIS + TERADATA
7. MAINFRAMES +TERADATA
8. FSLDM DEVELOPER[financial Services Logical Data Model] (Banking, insurance)
9. MLDM DEVELOPER[Manufacturing " "]

DEFINATION:-

IT IS AN RDMS (1979) THAT DRIVES COMPANY'S DATAWAREHOUSE OLAP, OLTP, DSS, APPLIANCES. (TDB)

A) AN "OPEN SYSTEM" WHICH EXECUTES ON UNIX MP-RAS (OR) WIN 2000, XP etc and above O.Ss

B) COMPATIBLE WITH ANSI STANDARDS

C) RUNS ON SINGLE OR MULTIPLE NODES (SERVER)

D) ACTS LIKE A SERVER

E) BUILT IN PARALLELISM

A) OPEN SYSTEM < platform independent
satisfy industry standards

D) SQL + PL = PLSQL[client]

SQL + SERVER = SQL SERVER

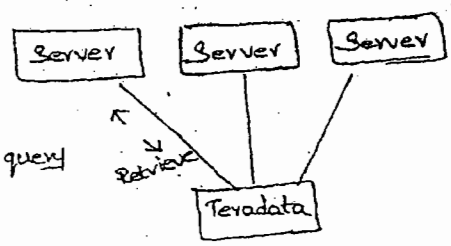
d)

6-Min		
7-Min	7 Min	3 queries Execution time
6-Min		
19 Min		
Oracle	Teradata	

9/12

1 TERADATA = 1024 GB = 10^{12}

e)



FAB

Small Company database → 2 to 3 Tera Bytes

Large Company database → 3 to 4 Tera Bytes

ETL - E - Extract → Getting the data

T - Transform → Doing intermediate Operations

L - Load → Loading to destinations.

Ex ETL TOOLS :-

DATASTAGE, INFORMATICA, ABINITO, SSIS ETC;

Teradata UTILITIES :- Loading & Unloading the data

(or)

Exporting & importing the data

ROLES :-

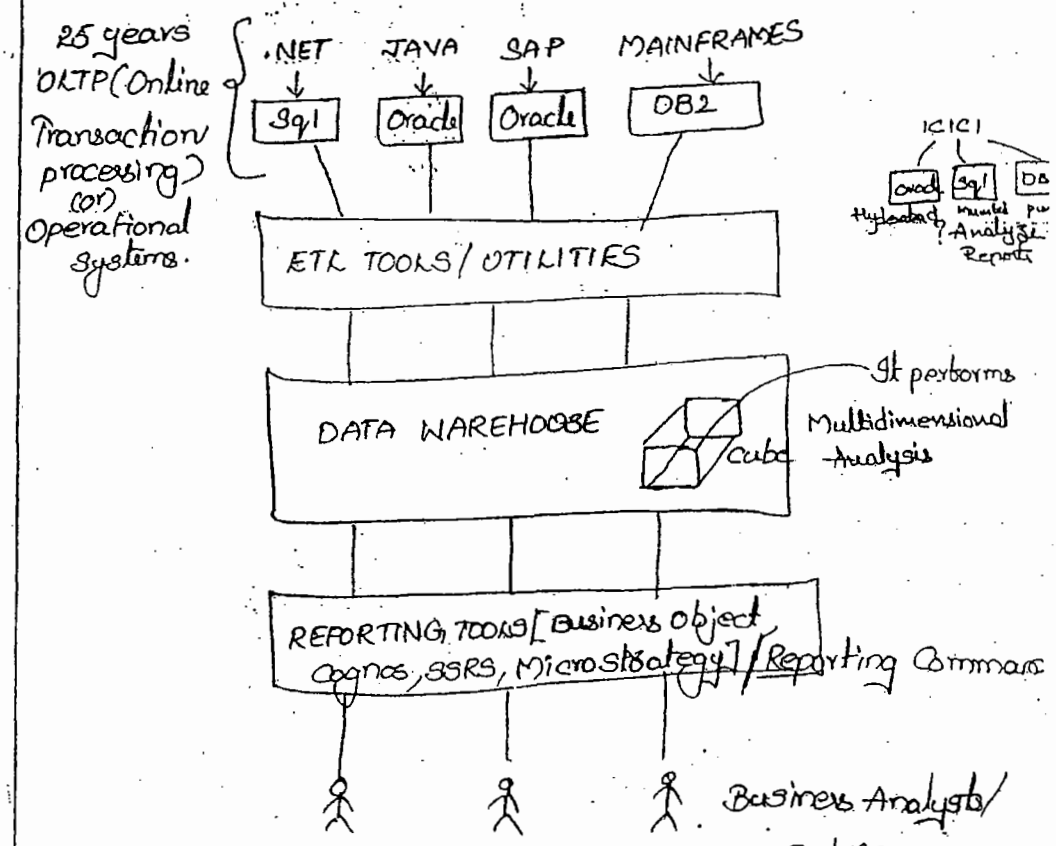
	Developer	DBA	
Oracle/Sql	60%	40%	
Teradata	80%	20%	because it provides

good Security and protection, good optimization techniq and better performance.

9/12

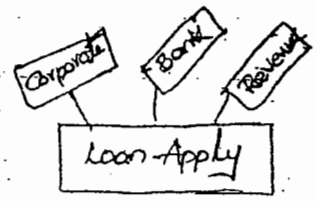
APPLIANCES :- It is not an application but it provides servic to the application.

TERADATA IN CURRENT ENTERPRISE :-

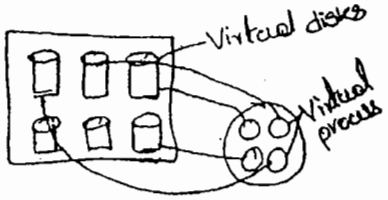
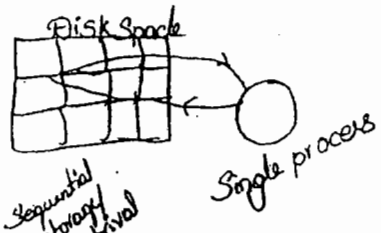


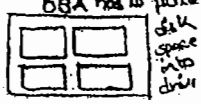
provides

DATA EVALUATION STAGES:-

		Eg:	Timetaken	No of rows accessed
TRADITIONAL SYSTEMS	OLTP [Online Transaction processing]	Immediate operations Ex: withdrawal cash from ATM.	Secs	Less no of rows
	DSS [Decision Support System]	Minor Analysis Ex:- the Sales diff between blue, green shirts in the year 2009, 2010.	Min/hrs	More rows.
MODERN SYSTEMS	OLCP [Online Complex processing]		Mins/hr/ Days	More rows.
	OLAP [Online Analytical processing]	Major Analysis [Multidimensional Analysis] Ex: the Sales diff between blue, yellow, red shirts in the years 2009, 2010, 2011, every year, every month, every location	Mins/hr/ days	More rows.

DIFFERENCES BETWEEN TERADATA & OTHER RDMS [Oracle, Sql Server]:-

TERADATA	OTHER RDMS
ARCHITECTURE: SHARED NOTHING 	SHARED EVERYTHING  By Sequential storage retrival

PROCESSES: MIPS [Millions of instructions /SEC]	KIPS.
STORES: BILLIONS OF ROWS [TERABYTES]	MILLIONS OF ROWS [GIGABYTES]
PARALLELISM: UNCONDITIONAL → Default it is drives. 	CONDITIONAL 
INDEXES: BETTER STORAGE & FAST RETRIEVAL 1st - 2, 2nd - 6, 3rd - 9	FAST RETRIEVAL ONLY 5th - 19, 6th - 22, 1st - 2, 2nd - 6
BULKLOAD FACILITIES : MANY → loading data depend on situations i.e, cursors, Triggers, constraints x, y	LIMITED → SQL * loader, SSIS (or) DTS
DESIGNED FOR: DWH, DSS, OLAP, OLTP	OLTP. More, OLAP LESS

* → Sql*loader is load the large volume of data & SSIS (or) DTS in a Sql Server.

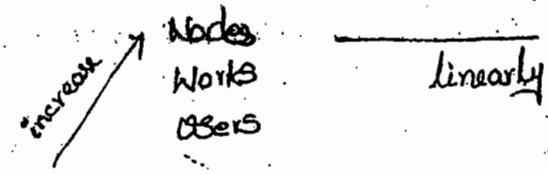
DTS - Data transform Service, SSIS - Sql Server Integrity Service

TERADATA COMPETATIVE ADVANTAGES:-

1. Automatic, Even Data Distribution:
   - parallel
   - uniform
   - random.

In Teradata the even, parallel, uniform, random distribution is Automatic.

2. HIGH SCALABILITY:-



Even we increase the no of nodes or work or users Tera data doesn't sacrifice any performance and it scales linearly

3. MATURE OPTIMIZER:-

Oracle Supports

	Latest Version
32 Joins/Query	64 Joins
32 ~~Joins~~ Subqueries/Q	64 Subqueries for a query

Teradata Supports

a) 64 Joins/query
b) 64 Subquery for a query
c) Aggregate operations
d) OLAP Operations with the help of powerball Optimizer.

4. MODELS THE BUSINESS:-

Teradata supports all business models like Star Schema, Snowflake Schema, Hybrid Schema, Bus Schema, Normalization, Galaxy etc,

* Upto 1999 Oracle & Sql doesn't have above features.

5. LOW COST TCO (TOTAL COST OF OWNERSHIP):-

Easy to install
Easy to work
Easy to manage
Many wizards
Cheaper price.

6. ACTS LIKE SINGLE DATASTORE:-

Teradata supports both channel Attached as well as Network Attached Systems.

Channel Attached System [Mainframes, COBOL, DB2 ...] Network Attached System [.Net, Java, Unix, Windows]

Teradata

7. Many bulk load facilities:-

BTEQ
FASTLOAD
MULTILOAD
TPUMP
FAST EXPORT
TERADATA SQL ASSISTANT [QUERY MAN] - - - -

X, Y, Z (refer BULKLOAD FACILITIES)

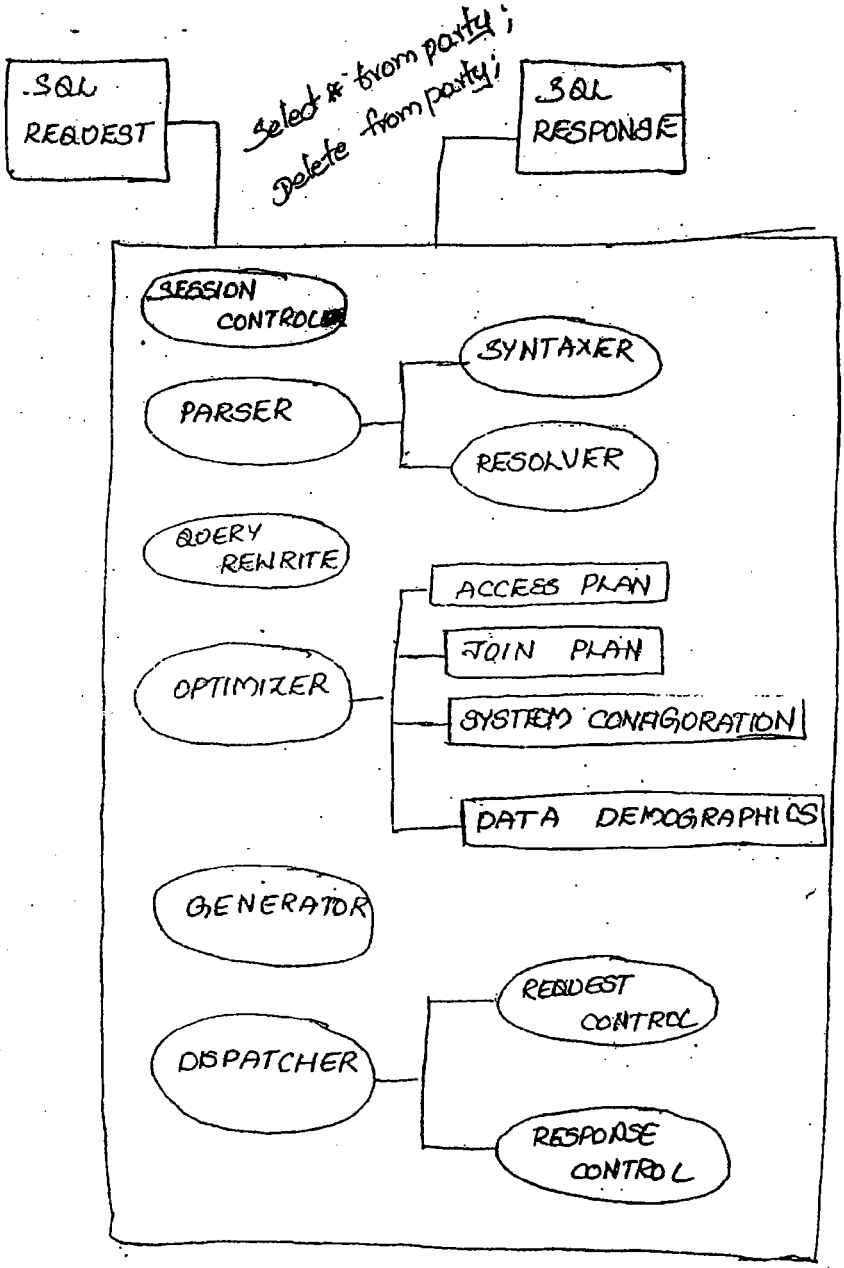
8. DESIGN FOR ANALYTICAL APPLICATIONS AS WELL AS TRANSACTIONAL APPLICATIONS:-

13/12

TERADATA IMPORTANT COMPONENTS:-

1. PE 2. BYNET 3. AMP 4. VDISK

1. PE [PARSING ENGINE]:-



Teradata 120 Users } Session control
Oracle 60 Users }

Master data is not updated into master data tables but it is stored in ODS tables only.

Press F3, F3, F3

⊙ Data base table /BIC/XSHID_KK (Navigational SID table)
ATTR

Select 'display'
'Contents'
'Execute'

Data is not updated into X-table

Press F3, F3, F3

Part-III: Demonstration on overwrite functionality of ODS

Step-1: Modify the file content

Open your flat file

56 ROYAL 72 RANCHI 555555

Save the file

E: close the file.

P-2: Reschedule the Infopackage

Select your Infopackage

Context menu

Select 'schedule'

Select 'Start'

P-3: Monitor the data

check data in PSA

check the data in ODS tables

select 'manage data targets'

Select 'Contents' tab

Select 'Active data' table

Select Execute

NOTE: ~~Contains data;~~ ~~opt~~ Observe the columns with cyan colour (Addr, Phone) indicates that they are Navigational ATTR
Press F3, F3.

Select 'Contents' tab

Select 'New data' table

select 'Execute

contains no data after activation

Press F3

Select 'Active data' table

Select 'Execute'

contains ATTR data after activation

NOTE: Select the record to be modified to demonstrate the overwrite functionality of ODS.

S6 Goyal 27 Patna 0423423423

Press F3, F3.

Select 'change log table'

Select 'Execute'

Observe the record mode

N → New image

Press F3, F3

NOTE: check the data in the underlying data dictionary tables

- (PSA, ODS, MD tables)

Enter T-Code : SE11

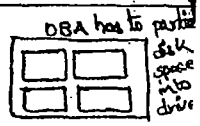
Master data tables:

1) ⊙ Data base table : /BIC/PSMID_KK

Select 'display'

Contents

Execute

PROCESSES: MIPS [Millions of instructions /SEC]	KIPS.
STORES: BILLIONS OF ROWS [TERABYTES]	MILLIONS OF ROWS [GIGABYTES]
PARALLELISM: UNCONDITIONAL → Default it is drives.	CONDITIONAL 
INDEXES: BETTER STORAGE & FAST RETRIEVAL 1st - 2, 2nd - 6, 3rd - 9	FAST RETRIEVAL ONLY 5th - 19, 6th - 22, 1st - 2, 2nd - 6
BULKLOAD FACILITIES : MANY → loading data depend on situations i.e, cursors, Triggers, constraints X, Y	LIMITED → SQL * loader, SSIS (or) DTS
DESIGNED FOR: DWH, DSS, OLAP, OLTP	OLTP. More, OLAP LESS

* → Sql*loader is load the large volume of data & SSIS (or) DTS in a Sql Server.

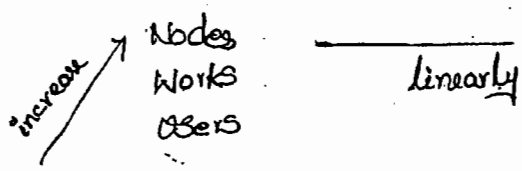
DTS - Data transform service, SSIS - Sql Server Integrity Service

TERADATA COMPETATIVE ADVANTAGES:-

1. Automatic, Even Data Distribution:
   - parallel
   - uniform
   - random.

In Teradata the even, parallel, uniform, random distribution is Automatic.

2. HIGH SCALABILITY:-



Even we increase the no of nodes or work or users Tera data doesn't sacrifice any performance and it scales linearly

3. MATURE OPTIMIZER:-

Oracle Supports

	Latest Version
32 Joins/Query	64 Joins
32 ~~Tera~~ Subqueries/Q	64 Subqueries for a query

Teradata Supports

a) 64 Joins/query
b) 64 Subquery for a query
c) Aggregate Operations
d) OLAP Operations with the help of powerball optimizer.

4. MODELS THE BUSINESS:-

Teradata Supports all business models like Star Schema, Snowflake Schema, Hybrid Schema, Bus Schema, Normalization, Galaxy etc;

* Upto 1999 Oracle & Sql doesn't have above features.

5. LOW COST TCO (TOTAL COST OF OWNERSHIP):-

Easy to install
Easy to work
Easy to Manage
Many wizards
cheaper price.

6. ACTS LIKE SINGLE DATASTORE:-

Teradata Supports both channel Attached as well as Network Attached Systems.

Channel Attached System [Mainframes, COBOL, DB2 --] Network Attached System [.Net, Java, Unix, Windows]

7. Many bulk load facilities:-

BTEQ
FASTLOAD
MULTILOAD
TPUMP
FAST EXPORT
TERADATA SQL ASSISTANT [QUERY MAN] - - - -

X, Y, Z (refer BULKLOAD FACILITIES)

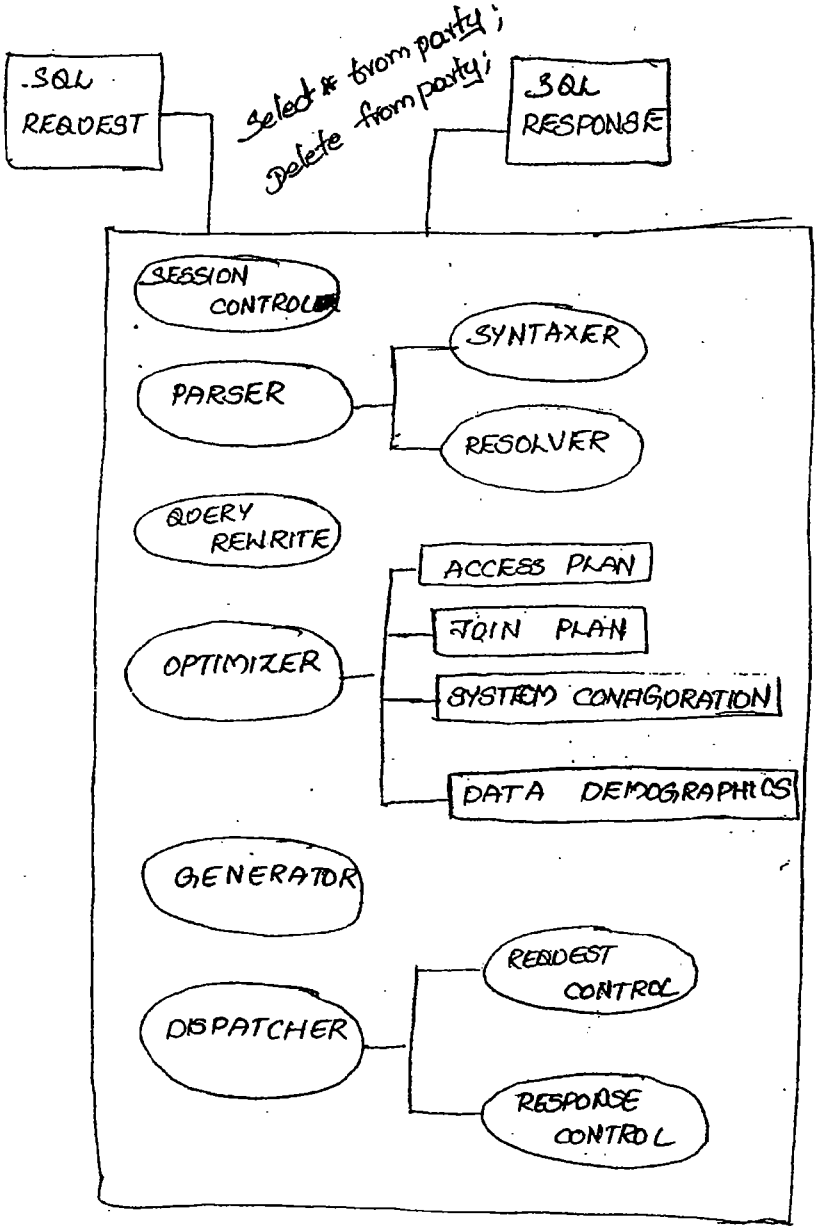
8. DESIGN FOR ANALYTICAL APPLICATIONS AS WELL AS TRANSACTIONAL APPLICATIONS:-

13/12

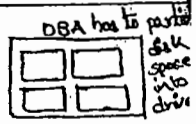
TERADATA IMPORTANT COMPONENTS:-

1. PE 2. BYNET 3. AMP 4. VDISK

1. PE [PARSING ENGINE] :-



Teradata 120 Users } Session
Oracle 60 Users } control

PROCESSES: MIPS [Millions of instructions /SEC]	KIPS.
STORES: BILLIONS OF ROWS [TERABYTES]	MILLIONS OF ROWS [GIGABYTE
PARALLELISM: UNCONDITIONAL → Default it is drives. 	CONDITIONAL 
INDEXES: BETTER STORAGE & FAST RETRIEVAL 1st - 2 2nd - 6 3rd - 9	FAST RETRIEVAL ONLY 5th - 19 6th - 22 1st - 2 2nd - 6
BULKLOAD FACILITIES : MANY → loading data depend on situations i.e, cursors, Triggers, context X, Y	LIMITED → SQL * loader, SSIS (or) DTS
DESIGNED FOR: DWH, DSS, OLAP, OLTP	OLTP. More, OLAP LESS

* → Sql*loader is load the large volume of data & SSIS (or) DTS in a Sql Server.

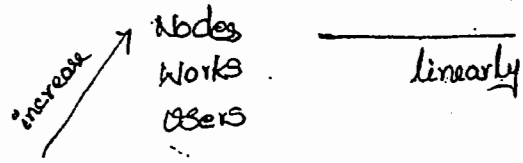
DTS - Data transform service, SSIS - Sql Server Integrity Service

TERADATA COMPETATIVE ADVANTAGES:-

1. Automatic, Even Data Distribution:
   - parallel
   - uniform
   - random.

In Teradata the even, parallel, uniform, random distribution is Automatic.

2. HIGH SCALABILITY:-



Even we increase the no of nodes or work or users Tera data doesn't sacrifice any performance and it scales linearly

3. MATURE OPTIMIZER:-

Oracle supports

	Latest Version
32 Joins/Query	64 Joins
32 ~~Tera~~ Subqueries/Q	64 subquiees for a query

Teradata supports

a) 64 Joins/query
b) 64 Subquery for a query
c) Aggregate operations
d) OLAP operations with the help of powerball optimizer.

4. MODELS THE BUSINESS:-

Teradata supports all business models like Star Schema, Snowflake Schema, hybrid Schema, Bus Schema, Normali-tion, Galaxy etc,

→ 1999 Oracle & sql doesn't have above features.

Interprets Sql requests and responses.

1) Session Control:-

A Session is Nothing but logical Connection between User and Application. Here it Controls the authorisation if it is Valid it does Logon otherwise it does Logoff.

2) ParSer:-

A) It Checks Synthatical Errors [; Missing at end of the statement]
B) It Checks Symantical Errors [Select from * Party;]
C) It Checks Existence of Objects

4/12

3) Optimizer

It displays the execution plan of the Sql Statement that is going to be processed in the Teradata Server.

Execution plan

It uses the below Components while displaying the plan

A) System Configuration
B) Available parallel Units
C) Access plan, Join plan etc;

4) Generator:-

It generates Steps for the plan provided above

5) Dispatcher:-

It takes set of requests and keep in a queue and delivers set of response by keeping codes same queue that means it does requests, responses flow control

Session control

NOTE:-

PE handles max 120 User Sessions.

2. BYNET [BANYAN NETWORK]:-

It Acts like "Message Communication" Layer between Components.

1. SMP BYNET [PE-AMP]:
   - POINT-POINT – One msg from PE – One Amp
   - MULTICAST – " " – many Amps
   - BROADCAST – " " – all Amps

2. MPP BYNET [NODE-NODE]:
   - POINT-POINT – One msg from one Node to another Node
   - MULTICAST – One msg from one Node to Many Node
   - BROADCAST – One msg from one Node to All Nodes.

3. AMP [Access Module Processor]:-

AMP is a Virtual processor and responsible for managing a portion of the database with in the base collection of Virtual disks.

This portion is not Sharable by any other Amp. So we called this Architecture as Shared Nothing Architecture.

Amp Contains Database Management Subsystem and it performs below operation

A) DDL
B) DML
C) Applying and releasing the locks
D) Joins, Aggregations etc;

4. V1

5. LOW COST TCO (TOTAL COST OF OWNERSHIP):-

Easy to install
Easy to work
Easy to Manage
Many wizards
Cheaper price.

6. ACTS LIKE SINGLE DATASTORE:-

Teradata Supports both channel Attached as well as Network Attached Systems.

Channel Attached System [Mainframes, COBOL, DB2 --] Network Attached System [.Net, Java, Unix, Windows]

7. Many bulk load facilities:-

BTEQ
FASTLOAD
MULTILOAD
TPUMP
FAST EXPORT
TERADATA SQL ASSISTANT [QUERY MAN] - - - -

X, Y, Z (refer BULKLOAD FACILITIES)

8. DESIGN FOR ANALYTICAL APPLICATIONS AS WELL AS TRANSACTIONAL APPLICATIONS:-

TERADATA VERSIONS:

- - DATABASE COMPUTER / CONTROLLER

The first DBC/1012

TDV1 R1 — V1 R4

[V1R1, V1R2, V1R3, V1R4]

TDV2R1 — V2R7

TD 12

TD 13 View Point

TD 14 Upcoming 2011

famous Versions: V2R5 & TD12

↓

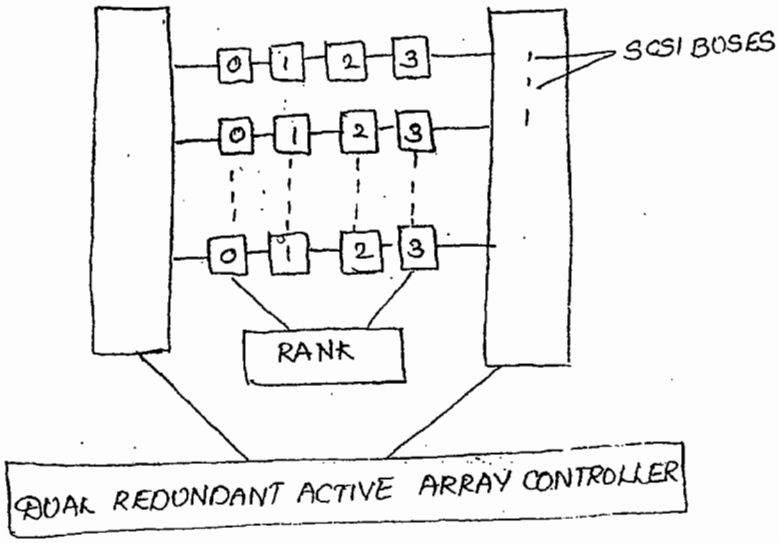
Till 2010 April

Certifications

Certificate Verification (new): TD12

11/12

4. VIRTUAL DISK :-



Collection of physical disks or Cylinders are arranged in Array fashion is called Virtual Disk or VDisk.

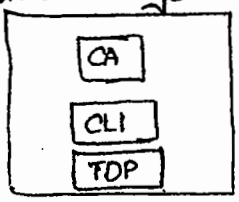
Traditionally this is called as Disk Array or Array of Disk.

After Amp performing the operations it invokes the controllers to store and manage the data access the Cylinders.

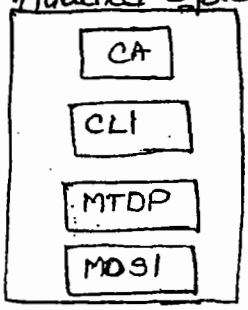
Here the two Controls are active always. So we called these controllers as DUAL REDUNDANT ACTIVE ARRAY CONTROLLER.

The Number of physical disks in an Array is called RANK. As per the diagram the Rank is 9.

Channel Attached System



Network Attached System



15/12

TDP → TERADATA DIRECTORY PROGRAM

MTDP → MICRODATA " "

CLI → CALL LEVEL INTERFACE

MOSI → MICRO OPERATING SYSTEM INDEPENDENCE

CA → CLIENT/CHANNEL APPLICATION

TDP:-

It is the high level component which does the below operations.

a) start creating sessions.

b) stoping the sessions.

c) It balances multiple sessions.

MTDP:-

It is just like TDP but cannot balances the sessions.

MOSI:-

It checks the platform indpendence of the Operating System.

CLI :-

It is a low level interface which does request response blocking and unblocking.

CA :-

client or channel Application.

15/12

TERADATA PRACTICALS :-

TERADATA SERVER STARTING :-

1. GO TO DESKTOP → TERADATA SERVICE CONTROL → START

(OR)

2. GO TO SERVICES → TERADATA DATABASE INITIATOR → START

NAVIGATIONS :-

DATASOURCES :- START → SETTINGS → CONTROLPANEL → ADMINISTRA

→ TIVETOOLS → DATASOURCES

SERVICES :- START → SETTINGS → CONTROLPANEL → ADMINISTRA

→ TIVETOOLS → SERVICES.

SERVER RUNNIG STATUS :-

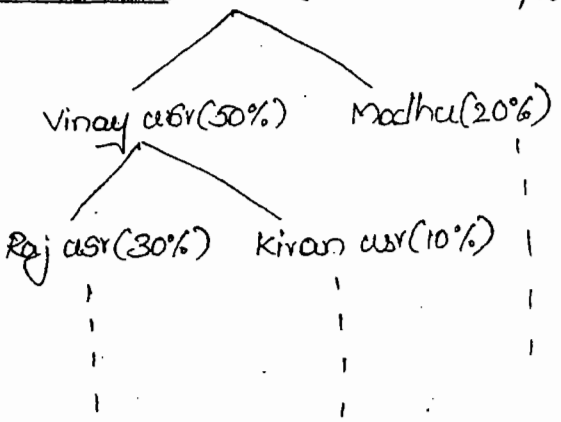
ORANGE ← GREEN → Running

GREY ← RED → STOPPING

TERADATA QUERY SUBMITTING TOOLS:-

TERADATA SQL ASSISTANT (QUERYMAN)	BTEQ(BASIC TERADATA QUE -RY)
NAVIGATION; 1STWAY : START→ RUN→ QUERYMAN(IN TO 13 SQLA)	1. START → RUN → BTEQ
2NDWAY; START → PROGRAMS → TDSQLASSISTANT	2. START → PROGRAMS → TERADA -TACLIENT → TERADATA BTEQ
1. WIZARD BASED OR GUI BASED	COMMAND MODE
2. WE CAN VIEW THE HISTORY HERE	N/A
3. WE CAN SET COLORS, THE CHARACTERS FOR DISPLAYI -NG NULLS, RUNNING MULTIPLE QUERIES AT A TIME ETC. - -	N/A
***4. MAINLY DESIGNED FOR CONNECTING AND FIRING ANY ODBC COMPLAINT DATABASE (ORACLE, SQL SERVER, ETC. -) RELATED QUERIES AND COMMANDS	MAINLY DESIGNED FOR TERADATA RELATED QUERIES AND COMMANDS.

DEFAULT USER/DATABASE: DBC (Database Computer 100%)



	UID	PSWD
ADMINISTRATOR:	DBC	DBC
TERADATA USER:	tduser	tduser
Workload Manger:	tcom	tcom

If we install any s/w the default IP address is

127.0.0.1.

DATA DICTIONARY:-

The Data Dictionary

- is an integrated set of system tables.
- Contains definations of and information about all objects in the system
- is entirely maintained by the RDMS
- is 'data about the data' or 'metadata'
- is distributed across all AMPs like all tables.

- May be queried by administrators or support staffs
- is accessed via Teradata Supplied Views.

Examples of Views:-

DBC.Tables - info about all DATABASES, USERS, AND THEIR CORRESPONDING OBJECTS.

DBC.AllRights - info about access rights

DBC.AllSpace - info about space utilization

DBC.COLUMNS - info about all COLUMNS.

DBC.ERRORMSGS - info about all ERRORS AND THEIR CODES.

DBC.DISKSPACE - info about DISKSPACES.

DBC.TABLESIZE - info about tablesizes

Working with Views:-

Eg: Select * From DBC.Tables.

DEFAULT DRIVERS:-

A) TDADMIN
   - USID: DBC
     PSWD: DBC

B) TDUSER
   - USID: tduser
     PSWD: tduser

C) TWMUSER
   - USID: twm
     PSWD: twm

t staff

AND THEIR

CODES. 17/12

WORKING WITH QUERYMAN:-

Open QueryMan

↓

Click Connect ODBC

↓

Machine Data Source

↓

Select tdadmin

↓

OK

1. Display the User Who Currently logged in?

SELECT USER (OR) SEL USER

Execute:-

O_2 (or) F5

Answer:-

User

1. DBC

2. Display the database, the User is Currently working with

SELECT DATABASE

Database

1 DBC

VIEW SHOW HISTOR:-.

It displays the history.

1. To do Certains at Queryman level we need to go for Tools→Option.

1a) General

↓

Display this String for Null data Values [?]

1b) Code Editor

Here we can change the colors, fonts, Tab sizes etc; for commands, keywords, comments etc;

1c) Query

The below Option is important and we need to enable after installation of Teradata s/w. otherwise all queries in the queryman will run.

☑ Submit Only the Selected query text only highlighted.

1d) Answerset

Maximum number of answer rows to display [100]

~~Hard~~

1e) Export/Import

Specify the format of data such as delimiter and Single code, double code while doing to processing

14) File Paths

These are the paths where we store, queries, Answerset and History Notes.

1. Display All database Users and they corresponding object (Tables, Views, Micros etc;) SELECT * FROM DBC.TABLES.

DATABASENAME	TABLENAME	TABLE KIND.
DB(or) USER NAME	Table	T
	View	V
	Procedure	P
	Macro	M
	Trigger	G
	Functionname	F

2. Display all tables inside the database VINAYAKA.

SELECT * FROM DBC.TABLES WHERE DATABASE='VINAYAKA'
AND TABLEKIND='T'

Note:- This is wrong Select * from Vinyaka.Tables.

3. Display all Users in teradata

SELECT * FROM DBC.USERS.

4. Display all Columns in Teradata

SELECT * FROM DBC.COLUMNS

5. Diaplay all Tables wheather the Columnname = partyid Exists.

SELECT * FROM DBC.COLUMNS WHERE ~~PART~~ COLUMNNAME = "PARTYID"

6. Display all Error Codes And Error Messages.

SELECT * FROM DBC.ERRORMSGS.

7. Display the table sizes.

SELECT * FROM DBC.TABLESIZES.

CREATING THE NEW DATABASE:-

1. A Database is passive Repository where we store all Database objects.
2. Untill youc create an object it is empty
3. It doesn't contain Any password.

Syntax:-

create database <databasename> from <parentdatabase> as

permanent = <memoryspace>,

spool = <memoryspace>,

temporary = <memoryspace>;

Ex:-

Create database db_DEVT from dbc as

permanent = 2000000, spool = 2000000, temporary = 2000000;

Select * from dbc.tables where databasename = 'db_DEVT'

Create table db_DEVT.test (partyid integer, partyname varchar(30))

15/12

CREATING A USER:-

A USER is a Active Repository where we store All database objects until you create a object it is empty.

Users contain password where As database doesn't contain password.

QUERY (tdadmin):- TERADATA SQL ASSISTANT (tdadmin)

Create User User_DBC from dbc

as

permanent = 2000000,

spool = 2000000,

Temporary = 2000000,

SELECT * FROM DBC.Tables where DATABASENAME = 'USER_DBC

CREATE TABLE USER_DBC.TEST [PARTYID INTEGER, PARTYNAME VARCHAR(30))

USER	DATABASE
→ CONTAINS PASSWORD	→ Doesn't
→ Active Repository Eg; 100 Request < 100 >100	→ PASSIVE REPOSITORY 100 Request = 100 Request
→ WITH THE USER WE CAN CONNECT TO ANY TOOL IN TERADATA	→ TO WORK WITH DATABASE WE NEED A USER.

NOTE:-

CREATING A DRIVER TO CONNECT TO THE OWN USERS SPACE

RUN → CONTROL PANEL → ADMIN TOOLS → OPEN DATASOURCES

↓

ODBC DATA SOURCE

↓

SYSTEM DSN

↓

CLICK ADD

↓

SELECT TERADATA

↓

FINISH

NAME; TERADATA BATCH → ODBC DRIVER SETUP FOR TERADATA DATABASE

IP Address : 127.0.0.1

UserName : USR_DBC

Password : Vinay

↓

OK

↓

Yes

Open Queryman

↓

click connect

↓

Select driver

So that Now we will be in your own user space call USER_DBC.

CONNECTING TO ANOTHER DATABASE OR USER:-

There are 2 ways.

(A) By specifying database name or USERNAME before the object.

Syntax:-

<DATABASE/USERNAME> <OBJECTNAME>

Eg:-

SELECT * FROM VINAYAKA.PARTY;

DELETE FROM DB_MADHO.TEXT;

(B) TAKE the database or user as your current database or user.

Syntax:-

DATABASE <DB/username>

Eg:-

DATABASE VINAYAKA

SELECT * from PARTY;

DELETE FROM TEST;

NOTE:-

From Take the data from database or other user u must have previliges on corresponding database or user. For providing for priviligies we use Administrative Tool.

Teradata Administrative Tool — RealTime no.

TERADATA SQL:-

→ 4th Generation Language

→ SET theory based Language

→ Contains set of other Languages.

1) DDL:- DATA DEFINATION LANGUAGE:

USE: For Creating, Altering and Removing objects

COMMANDS: CREATE, ALTER, DROP

NOTE: There is no Truncate Command.

2) DML:- DATA MANIPULATION LANGUAGE:

USE:- For Manipulating objects

COMMANDS: SELECT, INSERT, UPDATE, DELETE

3) DCL:- DATA CONTROL LANGUAGE:

USE: FOR providing And Revoking privilege for objects between users and databases.

COMMANDS:- GRANT, REVOKE, GIVE.

4) SPL:- STORED PROCEDURE LANGUAGE:

USE: CREATING and CALLING a stored procedure.

5) TERADATA EXTENSIONS TO SQL:

HELP

SHOW

EXPLAIN

COLLECT STATISTICS.

SHOW COMMAND:

It shows the DDC of the object

Syntax:

SHOW OBJECT <objectname>

Eg:

SHOW TABLE <TABLENAME>

SHOW VIEW <VIEWNAME>

HELP:

It describes the object from data dictionary.

Syntax:

HELP OBJECT <objectname>

Eg:

HELP TABLE <TABLENAME>

HELP MACRO <MACRONAME>

HELP PROCEDORE <PROCEDURENAME> ETC;

CREATE A TABLE:-

CREATE [SET/MULTISET] [GLOBAL TEMPORARY] VOLATILE]

TABLE <TABLENAME> <DATA OPTIONS>

<COLUMN DEFINATIONS> <CONSTRAINTS> <INDEXES>

SET TABLE: It doesn't allow duplicate records

1 vinay 10,000 X

1 vinay 10,000

MULTISET: Allows duplicate records.

FREESPACE: The amount of space free and assign from cylin-drical disk space.

DATA BLOCKSIZE: It is the physical input & output of the file system (MDF & LDF)

DATA OPTIONS:

FALLBACK, NOFALLBACK

BEFORE JOURNAL, NO BEFORE JOURNAL

AFTER JOURNAL, NO AFTER JOURNAL

DATA RECOVERY AND PROTECTION:

1. LOCKS : Prevents Simultaneous access on the objects

2. RAID : Prevents from disk failure.

3. FALLBACK : Prevents from Amp failure.

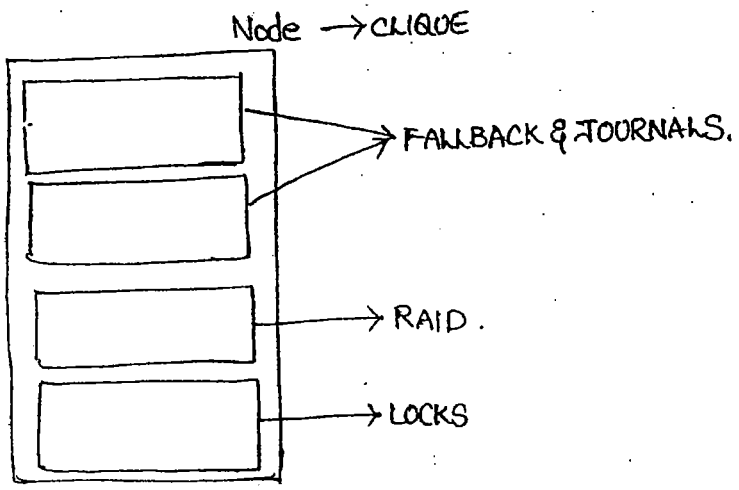
4. JOURNALS : Preventing images failure.

5. CLIQUE : Preventing Node failure

ARC [ARCHIVE AND RECOVERY] :-

A means of Archiving Data to TAPE AND DISK And Restoring Data TO Teradata database.

Backup Application SOFTWARE INCLUDES the following NETVAULT, NETBACKUP, TIVOLI STORAGE MANAGER TERADATA EXTENSION.



Here Every Level we have data recovery & protect

DATAWAREHOUSING FUNDAMENTALS:-

Informational System [OLTP System]	Dwh System
DATAIN	DATAOUT

Famous Authors INMON, KIMBALL, SEANKELLY.

DATAWAREHOUSE DEFINATION:-

Bill Inmon.

"A Data Warehouse is a

subject-oriented,
integrated,
nonvolatile,
and time variant

Collection of data in support of management's decisions."

SEAN KELLY

"the data in the data warehouse is:

Separate
Available
Integrated
Time stamped
Subject Oriented"

Datamart is → Specific func-task,
Dependent
Independent
Logical.

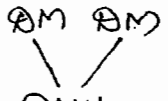
Creating a DATAMART :-

① Complex table
↓
Small table.

② Create a View
↳ it is datamart (DM)

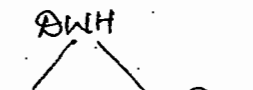
We construct warehouse in two Approaches.

Bottom up



KIMBOL

Topdown.



INMON

FAQ
** How do you recognize OLTP Database & DWH Database
In OLTP it is day to day operations it is normalized
In DWH Database is the denormalized

DATA MODELLING:-

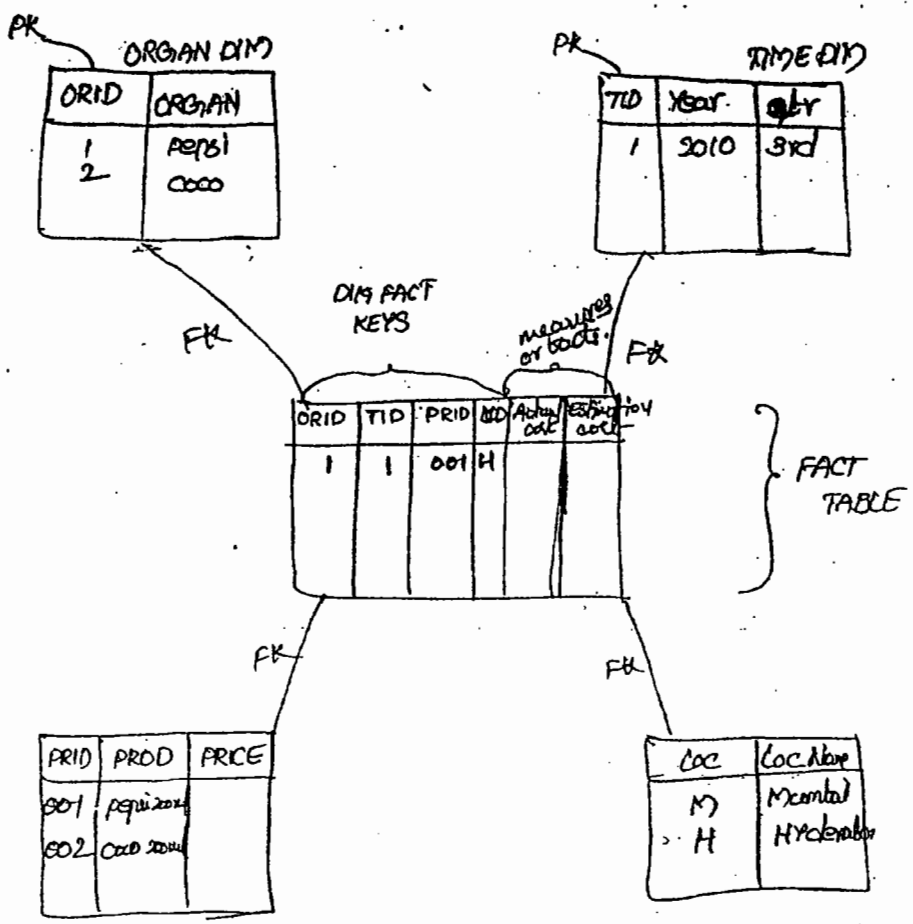
The way organising inside the database is called DATA MODELLING.

DATA MODELLING

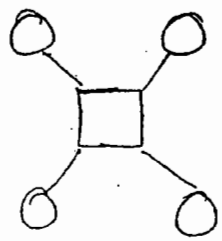
E-R Modelling	Dimension Modelling (good)
→ TABLE	→ DIMENSION
→ COLUMN	→ FACT
→ ATTRIBUTE	→ MEASURE
→ ROW ETC;	→ MEMBER ETC;
→ RELATIONSHIPS	→ RELATIONSHIPS.
1 – 1	Star Schema
1 – Many	Snow flake Schema
Many – Many	hybrid Schema.
→ OLTP APPLICATIONS	→ OLAP APPLICATIONS.

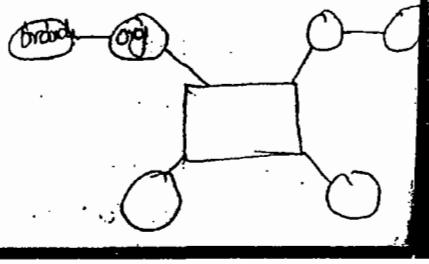
OLTP Database.
normalized.

DIMENSION TABLE	FACT TABLE
→ Specific Business info.	→ Measurable info
→ Textual info.	→ Numerical info.
→ No Measures	→ child table.
→ We use ask what/ where/when/why ?. you get the Answer	→ We ask how much/ how many ? you give the answers.



STAR SCHEMA	SNOWFLAKE
→ Dimensions directly connected to fact tables	→ Dimension connected to another dimension which in turn connects to fact table [Dimensional hierarchies are there]





→ Denormalized	→ Normalized
→ More memory	→ Less memory
→ Less joins: So great performance.	→ More joins So less perfo.
→ OLAP Systems [DWH Systems] More.	→ OLTP Systems more. OLAP Less.

NOTE:-

As Star Schema and DWH are denormalized so we take both as the combination in realtime.

NOTE:-

MAX 80 Dimension tables we can connect to fact table. we have multiple facts and Dimensions in Realtime.

Different Dimensions are:

- → Junk
- → Degenerated
- → conformed
- → Roleplaying
- → Slowly changing dimension.

→ Junk: dimensions are contains multiple branches into and maintain with some flags.

→ These are the dimensions which have values in the fact table but doesn't have description in dimension table.

e.g:- invoice number, order number

→ Confirmed :- It is shared dimension, multiple fact tables we can use.

→ Role Playing :- These dimensions plays diff roles in diff situations.

ex:- Customer table acts differently when work with current A/c & Saving A/c.

→ Slowly changing Dimension :- There are a three types to main the history.

Type 1 :- No history maintain

Type 2 & Type 3 :- History maintained.

Types of facts & Types of Measures :-

A) ADDITIVE : Added across all dimensions.

Eg: SUM, AVG OF SALES. SUM 120000

B) NON ADDITIVE : Added across None of the dimensions

Eg: %, RATIOS TOTAL GROWTH OF THIS YEAR % = 10

c) SEMI ADDITIVE : Added across some of the dimensions.

Eg :- FIRST QTR, SECOND QTR SALES.

FIRST QTR 40000

	Sales	growth
Jan →	10000	10%
Feb →	10000	10%
⋮	⋮	⋮
Dec →	10000	10%

ABLE:-

These Tables doesn't contains any measurable or Editable information.

Eg:- daily attendence to institute (there is no Additive measures)

CUBE:- It is multidimensional structure which consists of FACTS & DIMENSIONS either arise in a Star Schema or Snow flake Schema model.

Locks:-

There are four types of Locks available.

A) ACCESS

B) READ

c) WRITE

d) EXCLUSIVE

Default	USER A Applied	Other users can DO	
Select	ACCESS	ACCESS, READ, WRITE	→ Data inconsistency
Insert, update, delete	READ	ACCESS, READ.	→ Data consistency
	WRITE	ACCESS	
DDL	EXCLUSIVE	NONE	

ACCESS LOCK:-

When user Applied Access lock Other users can take the data and they can perform on required operations. So data inconsistency there.

READ LOCK:-

User A contains this Lock Other users can take the data. but they can't perform any operations. So data consistency there.

WRITE LOCK:-

One User is writing the data then other users just seeing the data and they cannot perform any operations

EXCLUSIVE LOCK:-

If users contain this Lock then No other user cannot do anything.

This Locks will be applied three levels.

A) Database level:- Here All objects inside database locked

Eg:- Tables, Macros.

B) Table level:- All rows inside the table will be locked

c) Row hash level:- Only the corresponding level row will be locked.

take
vations.

ke the
o data

users just
operation

cannot

free locked

he locked

row

Syntax :-

LOCKS AT ROW LEVEL;

SYNTAX : LOCKING ROW FOR <LOCKTYPE> <SQLQUERY>

LOCKING ROW FOR ACCESS SELECT * FROM PARTY;

LOCKING ROW FOR WRITE INSERT INTO PARTY VALUES(');

LOCKS AT TABLE LEVEL;

SYNTAX : LOCKING TABLE <TABLENAME> FOR <LOCKTYPE>

LOCKING TABLE PARTY FOR ACCESS;

RELEASING LOCK :

RELEASE LOCK DATABASENAME.TABLENAME;

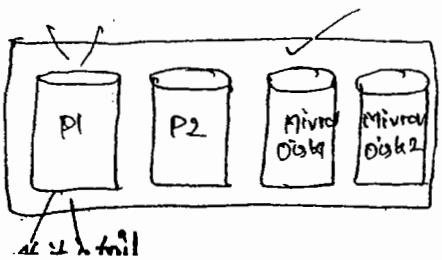
NOTE :-

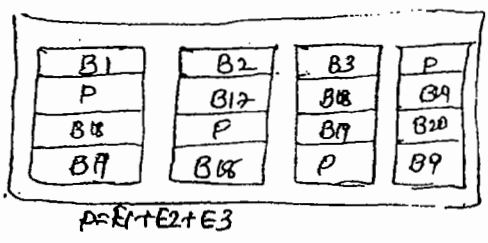
For Applying database lock we need teradata administrat
-ive tool.

RAID [REDONDANT ARRAY OF INDEPENDENT DISK DRIVES] :-

RAID0, RAID1, RAID2 - - - - RAID5, RAID6, RAIDE

RAID1 [Mirror Technique]	RAID5 [PARITY CHECKING TECH]

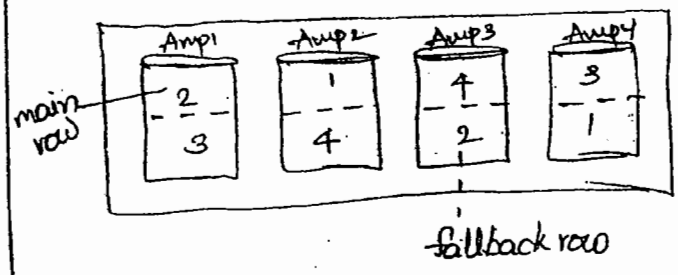




RAID1	RAID5
Here Every primary disk the mirror disk available. whenever primary disk goes down then Array Controller redirects the messages to the mirred disk and mirror disk performs the operations. In this way high availability is provide. Advantage:- Quick Recovery in case of Disk failure Drawback:- A) doubles the memory space it occupies. B) doubles the operations it performs..	For Every 3 blocks of data in three disk there is the parity in the 4th disk. So whenever the failure of any disk one can construct the blocks of info. with help of parity and remaining two blocks.. Advantage:- only 25% of addition memory it occupies. Recovery is slow drawback.

FALLBACK:-

Raid 1, 2, 3, 4



When we specified fallback at table level Every row that going's to distribute in the Amp will have the duplications in other Amp.

The Row is called as Fallback row & Tables is called as Fallback Table.

Here the advantage is whenever the main Amp goes down all instruction move to the fallback rows in other Amp. In this way its providing high availability.

Advantages:

Quick Recovery in terms of Amp failure.

Draw backs:-

Doubles the memory space it occupies & doubles the operations it performs.

NOTE:-

In Realtime many Companies use RAID1 & FALLBACK as there protection mechanism.

21/12

Fallback CLUSTER:-

A cluster is nothing but group of Amps here.

Here we have 2 to 16 Amps as a group.

Fallback cluster provides flexibility where if one Amp is down it goes to the fallback row in the same

group in other Amp.

If two Amps went down then RDMS Altered.

No. of Amps	Workload $1+\frac{1}{N-1}$	Remaing should Available (N-1
16	1.06	15
8	1.12	7
4	1.33	3
2	$1+\frac{1}{1}=2$	1

NOTE:-

On and Avg 4 Amps per Clusters is recomendable.

JOURNALING :-

It is nothing but a record which does some kind of activity.

There are 4 types of images support this journaling

A) Single image

One Copy of the data will be taken.

B) Dual image

Two Copies of the data will be taken.

a) Before Image

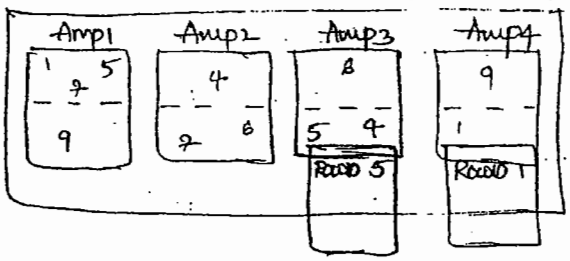
Before changes occured on the row the data will be taken.

d) After image

After changes happened on the row taking copy of data.

TYPES OF JOURNALS:-

1) Recovery Journal.
2) Transactional Journal (Transient Journal)
3) Permanent Journal



For Implementing recovery journal fallback should be enabled. whenever an amp goes down automatically recovery journal will be activated and it stores the ROWID's in the recovery row which are changed & belong to the down Amp.

USAGE:-

This journal usefull to recover the ~~main~~ down Amp well Once it comes online.

Transaction Journal:-

A Transaction is nothing but logical collection of stmts which can be either succeeded or failed.

USAGE:-

Here Transaction Journal help us successfull rollback in case of failure.

Successfull Transaction	Failed Transaction
Begin Transaction // Transaction Journal Enabled	Begin Transaction // Transaction Journal Enabled.
(S) Insert party; // Before image taken	(S) Insert Party; // Before image taken
(S) Update party;	(S) Update party; S-Success
(S) delete party;	(F) delete party; F-Failure
End Transaction	Successfull rollback happens
B.I Dropped	B.I repplied.
T.T Disable (or) Discarded.	B.I Dropped
	T.T discarded.

Note:-

These Transaction Journals & Recovery Journals are automatically called.

Permanent Journal:-

It does full or partial recovery.

It uses Before Images (B.I) & After Images (A.I) and performs the above operation.

B.I → Yesterday

A.I → Mon

⋮

AI → Sun

Full Recovery = B.I + (Mon - Sun) AI

Selective Recovery = B.I + (Mon + Tue) A.I

CHECKSUM:-

Levels of disk integrity checking in Teradata V2R5.1:

HIGH ↑ ABILITY TO DETECT ERRORS ↑ NONE

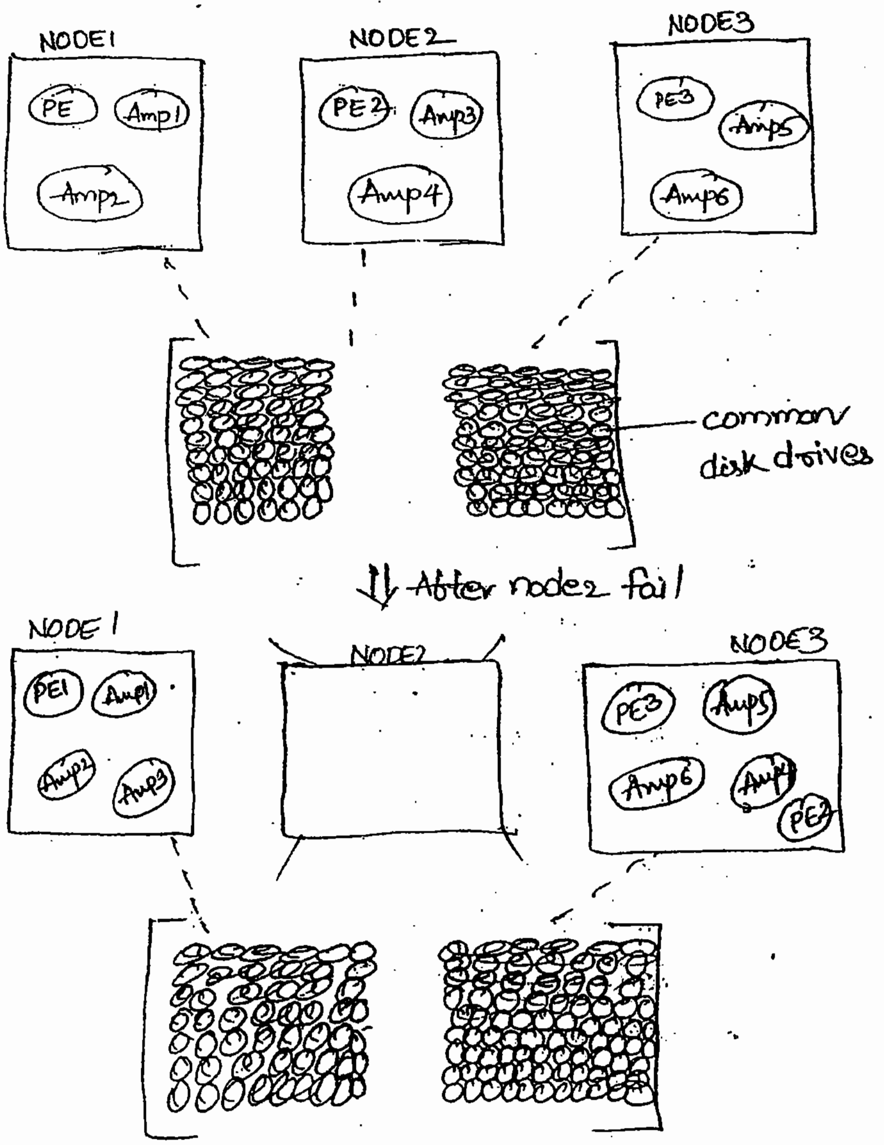
- ALL
  Full end-to-end checksums -- Detects most bit, byte, and byte string errors.
- LOW, MEDIUM, HIGH
  Statistically sampled partial end-to-end checksums -- Detects lost writes and intermediate levels of bit, byte and byte string errors.
- NONE
  Disable checksum disk I/O integrity checks -- Detect some forms of lost writes using standard file system metadata verification.
- DEFAULT
  When checksum is not specified - Defaults to the

setting for the table type set at the system level.

CLIQUE:-

It prevents from Node failure. here collection of Nodes will share common disk drive. whenever any node went down automatically virtual process will migrate from failed node to another node to retrive the data from common disk drives.

At a time 128 virtual processes will be active



CREATE TABLE:-

CREATE MULTISET TABLE TEMP, FALLBACK, NO BEFORE JOURNAL,
NO AFTER JOURNAL, CHECKSUM = DEFAULT
(PARTYID INTEGER, PARTYNAME VARCHAR(30)).

E5

NOTE:-
If we required Before journal & After journal at table
before that before journal & After journal will be
enabled at database or user level.

TABLE OPTIONS:-

A) SET TABLE

B) NO FALLBACK

C) NO BEFORE JOURNAL

D) NO AFTER JOURNAL

E) CHECKSUM = DEFAULT

F) PRIMARY INDEX(UPI(or)NUPI)

COLUMN DEFINATIONS:-

COLUMN: ATTRIBUTE

DATATYPE: TYPE OF DATA USING FOR A COLUMN

ANSI COMPLAINT DATATYPES	TERADATA EXPENSIONS
CHARACTER DATATYPES CHAR( ) VARCHAR( )	LONG VARCHAR( )...
NUMERIC DATATYPES INT SMALLINT BYTE FLOAT DECIMAL(M,N)	VARBYTE BYTEINT
GRAPHIC DATATYPES BLOB, CLOB	GRAFFIC, VARGRAPHIC.
OTHERS:-	DATE, TIMESTAMP, INTERVAL, PERIOD** (NEW IN TD13)

NOTE:-

Teradata supports ANSI COMPLAINTS as well as TERADATA Extensions.

DATATYPES ATTRIBUTES:-

These Attributes effects at the storage level and created only at table level creation.

DATATYPE FUNCTIONS	FUNCTIONALITY
TITLE	It takes title of the column.
UPPERCASE	stores the data in Upper case.
CASESPECIFIC	stores the data in specified case.
NOTNULL	Doesn't allow null values.
DEFAULT	Takes Users defined default value.
DEFAULT USER	It takes the users who currently logi
FORMAT	Takes the format for integer, string, date datatypes.
DEFAULT SESSION	It takes the session id of the current users who logged in.

Create table & inserting values :-

✓ CREATE SET TABLE DUM, FALLBACK, NOBEFORE JOURNAL, NO AFTER JOURNAL, FREESPACE=2 PERCENT, DATABLOCKSIZE= 10201 BYTES
(PARTYID INTEGER TITLE 'Company ID',
PARTYNAME VARCHAR(30) uppercase, partycode integer default 77 COMPRESS 100, jdate date format 'YYYY-MM-DD')

✓ sel * from dum

insert into dam values(1,'vinay',10,'2008-10-05');
insert into dam values(1,'vinay',10,'2008-10-05');
--duplicate error in terms of Set.
insert into dam values(2,'HDFC','2008-10-05');
insert into dam values(3,'hhvinay',20,'2008-23-10');
--failed because of Wrong date format.

27/12

COLUMN STORAGE ATTRIBUTE:-

COMPRESS:-

It compresses NULLS and USER DEFINED Values to Zero Space.

Ex:-

Pcode	Account 5CR
1Lakh	SB 4CR
90th ≡ } 90th-[x]	CB 1CV
	COMPRESS 'SB'

storing SB 4CR times with dist. addres its costing memory if we compress it it takes one address location in 4CR SB

1. It is used for memory saving. It compresses nulls and user defined values to zero space.
2. If we can compress multiple values (up to 255 columns plus null can be compressed).
3. It supports for fixed width text also.
4. INTEGER, BYTEINT, SMALLINT, DATE, DOUBLE, DECIMAL(1,2,4,8), CHAR(N), FLOAT/REAL(8) SUPPORTS.
5. VARCHAR, VARBYTE, VARGRAPHIC, TIME, TIMESTAMP, INTERVAL Not Supported.

Syntax:-

COMPRESS NULL OR <USER DEFINED VALUES>

Eg:- A. COMPRESS NULL

B. COMPRESS 'SAVINGS'

C. COMPRESS 'SAVINGS', 'CURRENT' (Multi Value Compression)

CONSTRAINTS:-

In Any database constraints are based on three follows.

1) Domain

NOT NULL

CHECK (Validates against a range)

DEFAULT

2) Entity

Primary key (Not allows Null Value) + (Unique Value)

Unique key (Allows One Null Value)

3) Reference

Foreign key

Constraints applied at two levels:-

a) Column level: On single column we take constraint.

b) Table level: On multiple columns (composite) we take constraints.

Syntax:-

COLUMNNAME DATATYPE CONSTRAINT <CONSTRAINT NAME> PRIMARY KEY/ UNIQUE/ REFERENCES/ CHECK.

Eg:-

```
CREATE SET TABLE DEPT   ---- parent Table.
{
DCODE INTEGER PRIMARY KEY NOT NULL,
DNAME VARCHAR(30),
DINCOME INTEGER CONSTRAINT CHK
CHECK(DINCOME >300000)
}
CREAT SET TABLE EMP   ---- child Table.
{
EID INTEGER UNIQUE NOT NULL,
ENAME VARCHAR(30),
DCODE INTEGER CONSTRAINT def
REFERENCES DEPT(DCODE))
```

TABLE LEVEL CONSTRAINTS:

```
CREATE SET TABLE DEPT1
{
DCODE INTEGER,
DNAME VARCHAR(30),
DINCOME INTEGER
} PRIMARY KEY(DCODE, DNAME)
```

SOFT REFERENTIAL INTEGRITY:

1. We can remove the data in the parent even the value existed in the child.
2. Implemented with NO CHECK OPTION.

```
CREATE SET TABLE EMP
{
EID INTEGER UNIQUE NOT NULL,
```

VARCHAR(3)

DCODE INTEGER CONSTRAINT REF

REFERENCES WITH NO CHECK OPTION DEPT(DCODE))

INDEXES:-

Data Distribution in other RDBMS:-

It is sequential because of this Reason it takes lot of time while storing and retrieval.

Data Distribution in TERADATA:-

A) AUTOMATIC:-

As it is Automatic it loads Any Volume of data.

B) EVEN:-

As it is Even it stores and retrieves the data parallely with maximum Speed.

CONCLUSION:-

This Automatic & Even distribution is possible in Teradata with the help of a concept called INDEXES.

INDEXES

PRIMARY INDEX [PE]

UNIQUE PRIMARY INDEX (UPI)

NON UNIQUE " " (NOPI)

PARTITIONED " " (PPI) < CASE_N / RANGE_N

NO " " (NOPI)

SECONDARY INDEX [SI]

UNIQUE SECONDARY INDEX [USI]

NON UNIQUE " " [NUSI]

Other INDEXES

JOIN INDEX [J.I]

HASH INDEX [H.I]

SPARSE JOIN INDEX [SJI]

VALUE ORDERED INDEX [VOI]

ETC;

PRIMARY INDEX :

1. It is the MANDATORY Index
2. Only 1 primary Index for the table.
3. A primary Index is Composite till 64 Columns.
4. It is a physical mechanism which assigns a row to the Amp.

5. It will be created at only a table level Creation
6. We Cannot Alter, Modify, Drop a primary Index (Unless you declare as NOPI)
7. If you take Unique Values then it is UPI.
8. If you take duplicates as Nulls then it is NUPI.

REALTIME USAGE:-

Generally in frequent used colums we go for PI.

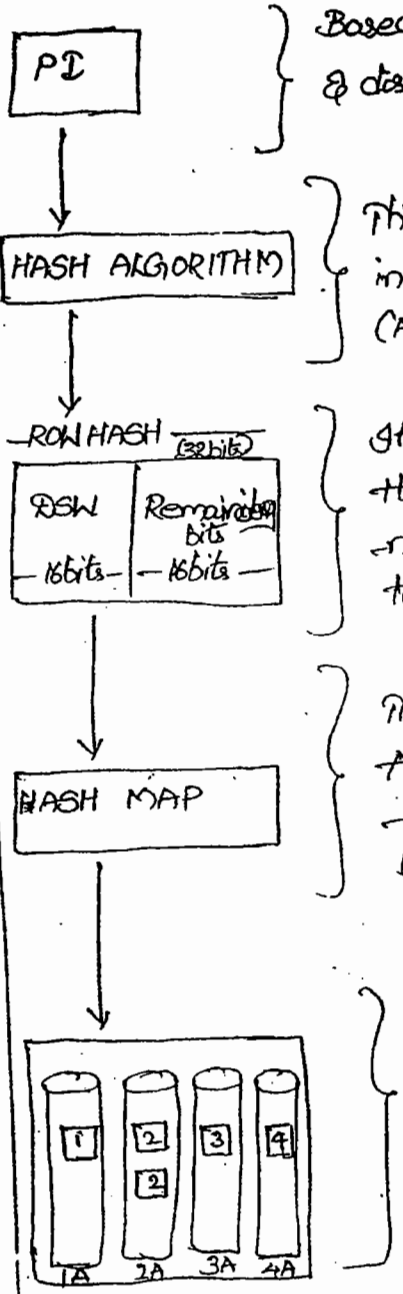
Eg:- CustomerId, Name, DOB in CustomerCare Table.

Primary Index Altering:-

There are two ways.

A) Drop the table & Recreate the table with New PI.
B) Create the table with NOPI if not able to decide proper PI on the table. Later Alter the NOPI to the proper PI.

PI PHYSICAL MECHANISM [storage & Retrieval Archi



Based upon these values the storage & distribution will happen

This is the common Algorithm the international character set is same (ASCI, EBCDIC, UNICODE ETC)

It is of 32bits generated from Hash algorithm. 1st 16bits are destination selection word which is used to identified Amp address.

This Map is same if number of Amps same in two different systems. It provides the relationship between DSW & Amp Address.

These are the Amp's where the Rows are distributed.

Destination selection word

1δ → 1A
2δ → 2A
3δ → 3A
4δ → 4A

Hash Collision:-

Storing the same row hash values or the same two primary index values is called Hash Collision. To prevent from these we go for the concept called ROWID.

ROWID:-

It identifies each Row Uniquely. It is It is of 64 bits Created at Amp level to different -iate One row to another row.

ROWID
64 bits

Row hash Value	System Generated Sequence value
32 bits	32 bits

2 — 25	1
2 — 25	2

UNIQUE PRIMARY INDEX:-

It takes only Unique Values.
It distributes the data across all amps Uniformly.

One Amp Operation and effects Only one Row

NOPI :- (~~Non Partitioned Primary Index~~)

Takes Duplicates & Nulls

Distributes the data across all amps with less uniformity.

Can also called as SKEWNESS. & Distribution is called SKEWED DISTRIBUTION.

One Amp operation & effects many rows.

****

PARTITIONED PRIMARY INDEX (PPI) :-

P.I → Table

PARTYID	TDATE
11	27/12
4	27/12
3	28/12
1	27/12
2	28/12
8	28/12
9	27/11

Row-hash	Tdate
1	27/12
4	27/12
9	27/12
11	27/12
2	28/12
3	28/12
8	28/12

1. DATA PARTITIONED ~~BASED ON TDATE~~

2. WITH IN THE PARTIONED DATA DISTRIBUTIED ON ROWHASH SEQUENCE

PARTITIONED PRIMARY INDEX:-

ROW_HASH	JDATE
1	27/12
2	28/12
3	28/12
4	27/12
8	28/12
9	27/12
11	27/12

1. DATA DISTRIBUTED BASED ON ROW_HASH SEQUENCE

ADVANTAGE PPI:-

1) In the situation the query goes to the corresponding partion and skips the other partion. So the performance is improved.
2) Mainly we used for range based data storing or category based data storing.

Syntax:- UPI:-

create table <tablename> (<columns>) unique primary index (colnames);

NUPI:-

Create table <tablename>(<columns>) primary index (colnames).

PPI:-

Create table <tablename>(<columns> primary index (columns names) partition by colname;

PPI:- No case:- Unmatched data goes to nocase
Unknow:- Null data goes to unknown.

Ex:-

UPI: CREATE SET TABLE test1
(
partyid INTEGER,
partyname VARCHAR(30))
unique PRIMARY INDEX(partyid);

NUPI: CREATE SET TABLE test
(
partyid INTEGER,
partyname VARCHAR(30))
PRIMARY INDEX(partyid);

PPI: CREATE SET TABLE test1
(
partyid INTEGER,
partyname VARCHAR(30), partyCode integer)
PRIMARY INDEX (partycode=10, partycode=20,
partition by CASE_N no case, unknown);

NOTE: WORKS ONLY WITH NUPI VALUES.

RANGE_N: Based on the range of data as part of partition we go for it.

CREATE SET TABLE test1
(
Partyid INTEGER,
Partyname varchar(30),
partycode integer, jdate date)
UNIQUE PRIMARY INDEX(partyid,jdate) partition by

RANGE_N (LDATE BETWEEN DATE '2010-01-01' AND '2010-10-01' EACH INTERVAL '1' DAY);

NOTE:- WORKS WITH DPI VALUES.

CREATE MULTISET TABLE tet (
partyid INTEGER, partyname Varchar(30))
NO PRIMARY INDEX;

Converting NOPI to PI table:-

There are 2 ways.

One way:

a) Create a multiset table with the same structure and take a PI on that table:

Create multiset table new_tet(partyid integer, partyname varchar(30)) primary index(partyid)

b) Insert data into new_tet, drop tet table and rename new_tet to tet

1. Insert into new_tet select * from tet.
2. Drop table tet
3. rename table new_tet to tet.

DEFAULT PRIMARY INDEX:-

If you don't create any primary index it takes below Scenario.

A) The P.K & U.K column taken as Unique primary index

B) If there is no primary key or unique key in table,

PRIMARY INDEX	PRIMARY KEY
Mandatory	optional in
physical Mechanism	Logical
Max 64 Column Combi -nation	No Limit.
Effects data distribution	Doesn't effect data distribu -tion.
Can't be dropped	Can be dropped
Can allow → Nulls → Duplicates.	Can't allow

** * Only one primary key for in One table in any database (oracle, Sql Server)

SECONDARY INDEX:-

1. It prevents full table Scan & provides Only fast Retrieval of data.
2. It uses a sub table mechanism for its operations
3. It doesn't distributes the data.
4. we can Create maximum 32 Secondary Indexes on the Table
5. A Secondary Index is Composite till 64 Columns.
6. It can be created and dropped at anytime. (at table creation or after table Creation)

If we take unique values then it is Unique Secondary index.

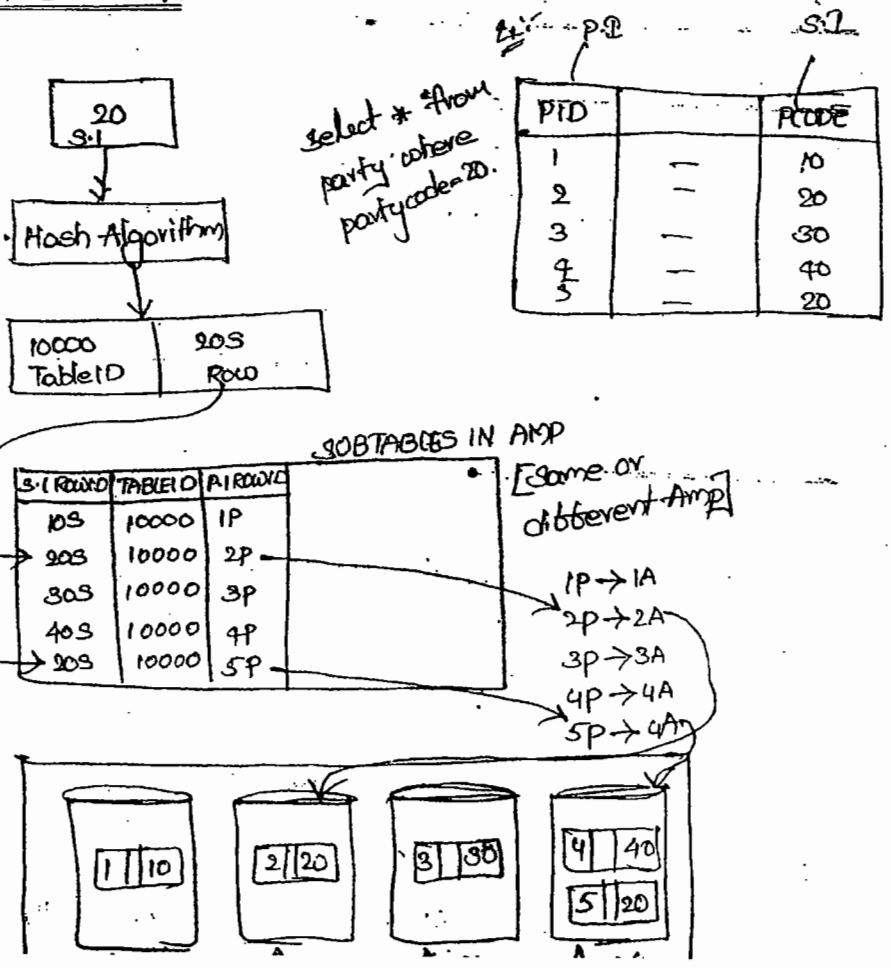
If we take duplicates & Nulls then it is NUSI.

9. If we take range base data by S.I Values sorting then go for value ordered. NUSI.

REALTIME USAGE:-

other than primary index columns if we required fast retrieval of data then go for S.I.

S.I SUBTABLE MECHANISM [STORAGE & RETRIEVAL ARCHITECTORE];-



29/12

USI (Unique Secondary Index):-

It takes only unique values.

Two Amp operation & Effects Only One row.

NUSI (Non Unique Secondary Index):-

It takes duplicates & Nulls.

Many Amp operation & effects many rows.

SI Creation & droping:-

At table level Creation:

USI:

Create table <tablename> unique index(colnames);

NUSI:

Create table <tablename> index(colnames);

After table Creation:

Syntax:

Create <unique> index <indexname> (colnames) on <tablename>

USI:

Create unique index idx_partycd(partycode) on party;

NUSI:

Create index idx_partycd(partycode) on party;

VALUE ORDERED NUSI:-

CREATE INDEX(PARTYCODE) ORDER BY VALUES ON PARTY;

Dropping indexes;-

Named index dropping:

drop index <indexname> on <tablename>

UnNamed index dropping:

drop index (colnames) on <tablename>

Final Example:-

create set table partytest, fallback
(partyid integer, partyname varchar(30) uppercase,
partyincome integer check(partyincome > 50000),
partycode integer)
unique primary index(partyid), unique index(partycode);

CREATING A TABLE FROM ANOTHER TABLE:-

1. CREATE TABLE <DATABASENAME>.<NEW TABLENAME> AS <DBNAME>.<OLDTABLENAME> WITH DATA --> With data

2. CREATE TABLE <DATABASENAME>.<NEW TABLENAME> AS <DBNAME>.<OLDTABLENAME> WITH NO DATA --> With No Data with only structure.

ALTER COMMAND SYNTAXES:-

ADDING COLUMN:

Syntax: ALTER TABLE <TABLENAME> ADD <COLUMNNAME> <DATATYPE>

Ex: ALTER TABLE PARTY ADD JDATE DATE

DROPPING COLOMN:

Syntax: ALTER TABLE <TABLENAME> DROP <COLUMNNAME>

Ex:- ALTER TABLE PARTY DROP JDATE

ADDING CONSTRAINT:

Syntax: ALTER TABLE <TABLENAME> ADD CONSTRAINT <CONSTRAINTNAME> PRIMARY/UNIQUE/CHECK/ REFERENCES.

Ex:- ALTER TABLE PARTY ADD CONSTRAINT CHK_SAL CHECK(PARTYSAL > 100000)

MODIFYING CONSTRAINT:

Syntax: ALTER TABLE <TABLENAME> MODIFY CONSTR -AINT . <CONSTRAINTNAME> PRIMARY/UNIQUE/ CHECK/ REFERENCES

Eg:- ALTER TABLE PARTY MODIFY CONSTRAINT CHK_SAL CHECK (PARTYSAL>100000 AND PARTYSAL<200000)

DROPPING CONSTRAINT:

Syntax: ALTER TABLE <TABLENAME> DROP CONSTRAINT <CONSTRAINTNAME>

Eg:- ALTER TABLE PARTY DROP CONSTRAINT CHK_SAL

NO FALLBACK TO FALLBACK:-

Syntax: ALTER TABLE <TABLENAME>, FALLBACK

Eg:- ALTER TABLE PARTY, FALLBACK.

CHANGING DATA BLOCKSIZE:

Syntax: ALTER TABLE <TABLENAME> DATABLOCKSIZE = <MEMORY> BYTES IMMEDIATE

Eg: ALTER TABLE <Table name>, DATABLOCKSIZE=1200 Bytes Immediate.

RENAMING THE COLUMN:-

Syntax: ALTER TABLE <TABLENAME> RENAME <OLDNAME> TO <NEWNAME>

Eg:- ALTER TABLE PARTY RENAME PARTYCD TO PARTYCODE;

DROP COMMAND: It Removes the Space & Data.

Syntax: DROP OBJECT <OBJECTNAME>

Eg: DROP TABLE <TableName>

DROP PROCEDURE <PROCEDURENAME>

DROP VIEW <VIEWNAME>

INSERT COMMAND:

1. INSERTING ONE SINGLE ROW:

INSERT INTO <TABLENAME> (col1, col2, -- , coln) VALUES (val1, val2, --- valn)

2. INSERTING FROM ANOTHER TABLE

INSERT INTO <TABLENAME1> SELECT */COLS FROM <TABLENAME2>

UPDATE COMMAND:

1. UPDATING ALL ROWS IN A COLUMN

UPDATE <TABLENAME> SET COLNAME = <VALUE>

2. UPDATING SPECIFIED ROWS.

UPDATE <TABLENAME> SET COLNAME = <VALUE>
WHERE <CONDITION> OR <SUBQUERY>

*** 3. UPDATING FROM ANOTHER TABLE

UPDATE <TABLENAME1> SET COLNAME> = <TABLENAME2>. <COLNAME> WHERE <TABLENAME1>. <COLNAME> = <TABLENAME2>. <COLNAME>

eg:

UPDATE employee SET phoneNO = phonelist.Number
WHERE employee.Name = phonelist.Name;

DELETE COMMAND: Removes DATA FROM THE TABLE.

Syntax:

DELETING SPECIFIED ROWS;

DELETE FROM <TABLENAME> WHERE <CONDITION>
OR <SUBQUERY>

DELETING ALL ROWS;

DELETE FROM <TABLENAME>

**
Question between diff Delete & Deleteall

DROP	DELETE
Removes Space [structure & data gone]	Only data removed.
No rollback	rollback is there.
DDL cmd	DML Cmd.
No where Condition	Where Condition taken
Dropping any object	Only table data it removes.

FAQ

Deleteall Works like truncate Command in other database.
It delete all the rows at a time where as normal delete will delete one by one Record at a time.

GRANT: Provides the specified access on the objects to the users

GRANT(ACCESS)

GRANT SELECT, INSERT, UPDATE, DELETE ON PARTY TO Peters -on, Jones with GRANT OPTION;

GRANT (ROLES):

GRANT BackupRole, AdminRole To SysAdmin, SysDBA With ADMIN option;

REVOKE:

REVOKE(ACCESS): Revokes the specified access on the objects from the Users.

REVOKE ALL BUT SELECT ON PARTY FROM USERA;

REVOKE(ROLE):

REVOKE ADMIN OPTION FOR Backups From Sysopi;

GIVE: Transfers the database Access Rights from One database to Another.

GIVE MyDB TO Finance;

ORDER BY CLAUSE:-

It SORTS THE DATA BASED ON THE SPECIFIED ORDER;

SYNTAX:

SELECT COL1, COL2, --- COLN FROM <TABLENAME>
ORDER BY COL1, COL2 --- ASC/DEC

Eg: A) SELECT PARTYID, PARTYNAME FROM PARTY
ORDER BY PARTYNAME

B) SELECT PARTYID, PARTYNAME, PARTYLOC FROM PARTY ORDER BY 3,2

In this QUERY. PARTYLOC SORTED FIRST: LATER PARTYNAME.

GROUP BY CLAUSE:-

It DISPLAYS THE DATA IN THE SPECIFIED GROUP AND

MAINLY USED TO DO AGGREGATE OPERATIONS.

SYNTAX: SELECT COL1, COL2 -- COLN FROM <TABLENAME>

GROUP BY COL1, COL2 --- (OR) 1, 2, 3 ---

Eg: DISPLAY THE PARTYINCOME SUM BASED ON THE PARTYLOC.

~~Ans: SELECT COL1, COL2 --- COLN FROM <TABLENAME>~~

~~GROUP BY COL1, COL2 --- (OR) 1, 2, 3 ---~~

~~HAVING <condition>~~

Ans: SELECT SUM(PARTYINCOME), PARTYLOC FROM PARTY

GROUP BY 2;

GROUP BY AND HAVING:

HAVING CLAUSE FILTERS THE DATA BASED ON THE GROUP SPECIFIED.

SYNTAX: SELECT COL1, COL2 --- COLN FROM <TABLENAME>

GROUP BY COL1, COL2 --- (OR) 1, 2, 3 ----

Eg: How DO YOU IDENTIFY DUPLICATES IN A TABLE

ANS: SELECT PARTYID FROM PARTY GROUP BY PARTYID

HAVING COUNT(*) > 1

INSERTING DUPLICATES IN TO ANOTHER TABLE:

INSERT INTO TABLENAME

SELECT * FROM TABLENAME2

WHERE COL8 IN (SELECT COL8 FROM TABLE2 GROUP BY 1 HAVING COUNT(*) > 1);

GENERATING SEQUENCE IN TERADATA:-

```
CREATE SET TABLE TEST
(PARTYID INTEGER GENERATED ALWAYS AS IDENTITY
(START WITH 1
 INCREMENT BY 1
 MINVALUE -2147483647
 MAXVALUE 2147483647
 NO CYCLE), PARTYNAME VARCHAR(30))
 PRIMARY INDEX (PARTYID);
```

DISTINCT CLAUSE:-

IT TAKES DISTINCT ROWS FROM THE QUERY

SYNTAX:

DISTINCT <COLUMNS>

EG:

```
SELECT DISTINCT PARTYID, PARTYNAME FROM PARTY
```

NOTE:

PERFORMANCE WISE GROUP BY GIVES GOOD PERFORMANCE OVER DISTINCT

HOW DO YOU ELIMINATE DUPLICATES IN A TABLE:

GO TO GOOGLE SEARCH THE ANSWER, THREE WAYS WILL BE VISIBLE

1. A. TAKE A NEW TABLE WITH SET OPTION
   B. LOAD THE DUPLICATED TABLE DATA, SO UNIQUE ROWS LOADED.
   C. DROP DUPLICATED TABLE AND RENAME SET TABLE NAME TO OLD TABLENAME

A. TAKE A NEW TABLE

B. LOAD THE DISTINCT ROWS IN TO IT, SO ONLOCE ROW LOADED.

C. DROP DOPLICATED TABLE AND RENAME NEW TABLE NAME TO OLD TABLE NAME

SELECT → WHERE

GROOP BY → HAVING

ROW_NOMBER
OR
RANK
} QOALIFY

REALTIME:

DISTINCT Clous Occupu more spool memory then go with Group By Having

PID	GROOP BY HAVING	Count	GROOP BY(PID) HAVING COONT(*)1	DISTINCT (PI
10	10	2	10	10
20	10			
30	20	2	20	20
10	20			
40	30	1	30	30
20	40	1	40	40
50	30	1	50	50

Inserting data in Above Test Table:-

INSERT INTO TEST(, 'VINAY');

O/P

PARTY ID	PARTY NAME
1	VINAY

CONDITIONAL EXPRESSIONS:-

IN: IT SEARCHES WITH IN THE LIST OF VALUES PROVIDED.

NOT IN: IT SEARCHES NOT IN THE LIST OF VALUES PROVIDED

EG: SELECT * FROM PARTY WHERE PARTYID IN(2,3,9)

EXISTS: IT FETCHES ~~OTHER~~ VALUES ~~WERE PROVIDED IF~~ THE EXISTED CLAUSE.

NOT EXISTS: IT FETCHES OTHER VALUES NOT PROVIDED IN THE EXISTS CLAUSE.

EG: SELECT * FROM PARTY WHERE EXISTS (SELECT PARTYID FROM PARTY).

IS NULL OR IS NOT NULL: IT CHECKS WHETHER THE, COLUMN CONTAINS NULL VALUES OR NOT.

EG: SEL * FROM PARTY WHERE PARTYNAME IS NULL.

BETWEEN AND: IT FETCHES THE DATA BETWEEN THE RAN -GE PROVIDED.

EG: SELECT * FROM PARTY WHERE PARTYINCOME BETWE -EN 20000 AND 50000

NOTE: INCLUDING 20000 and 50000

LIKE: IT FETCHES A FULL STRING FROM PARTIAL STRING

IT USES TWO WILD CHARD CHARACTERS.

A. % → FOR ALL CHARACTERS.

B. _ → FOR SINGLE CHARACTER

EG: DISPLAY THE NAMES WITH STARTS WITH 'V'.

SELECT * FROM PARTY WHERE PARTYNAME. LIKE 'V%'

OTHER OPERATORS WITH LIKE:

ALL: ALL CONDITIONS TO BE MET

SOME: SOME " "

ANY : ANY CONDITION "

EG: DISPLAY NAMES STARTS WITH 'V' OR STARTS WITH R

SELECT * FROM PARTY WHERE PARTYNAME LIKE SOME ('V%', 'R%')

NOTE:

Among IN and EXISTS, EXISTS GIVES good perform -ance than IN because EXISTS Stops the process once it finds the value.

TERADATA NAMING CONVENTION:

1. Each object Name can be maximum of 30 chara -cters.
2. A Name Can be combination of A-Z, a-z, 0-9, $, # and _.
3. No Name Should start with Numeric.
4. An object name must be unique in the database or user.

ARITHMETIC OPERATORS:-

+

-

*

%

[ ]

( )

LOGICAL OPERTORS:-

AND

OR

NOT

COMPARSION OPERATORS:

<

<=

>

>=

<> → Not Equals.

Order of Evaluation:-

→ Left to right

NOT, AND, OR

A and B Not C or D

BODMAS

[] ( ) % * + -

[2+3]/(4*5*[3*2]+6)

ARITHMETIC FONCTIONS:

ABC(argument) → absolute Value.

Sqrt(") → Square root

LOG(") → Log 10 Algorithm

LN (") → Log e "

RS:-

REALTIME OPERATIONS:-

Colarmn — Join

Roco — Set

Limfed access on tables - VIEWS

-function or operation — procedere.

Moving one place to another — Cursor.

unauthorized opera -tion — Trigger

SUB QUERIES:-

Based on one query result you can do operations in another query

SET OPERATIONS:-

It does operations in Roco wise.

JOIN'S OPERATIONS:-

It does operations in Columnwise.

VIEW'S:-

Providing Limited access on the table or tables.

PROCEDURE'S:-

Which does a Specific Business Operation or implemen-nt process.

FAQ

FONCTIONS:-

It does some operation but it returns some values.

CORSORS:-

It provides a flexibility by working with a rows in reverse set wit the below operations..:

A) going to first record.
B) " " last.
c) going to the Records one by one.

TRIGGERS:-

Provides Security on table to prevent unauth -orize insert, delete, update operations.

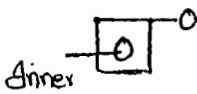
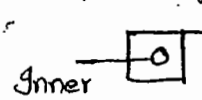
31/12
31

SUB QUERIES:-

Query inside another Query is called sub query.

Sub query types

- Simple Sub query
- Nested Sub query — Sub query inside another Sub query
- Corelated Sub query

Simple Query	Corelated Query
	
→ Inner Query executed first	→ inner query executed next first Outer query executa
→ Inner Query only one time	→ inner query executed many times based on Outer Query [like for loop]
Eg: finding 2nd maximum Select max(partyincome) from party where partyincome not in (select max(partyinc -ome) from party)	Eg: finding 2nd maximum Select a.partyincome from party a where 1=(Select count(distinct (b.partyin -come)) from party b where a.partyincome < b.partyincome) Note: For n-th max Salary need to place n-1 inplace of 1.

NOTE:-

We Need to eliminated Corealated Sub queires because it degrades the performance.

By forgetting the Answer for nth maximum Salary or top n salaries we need to use the below analy -tical follows.

1. RANK & Quality
2. ROW_NUMBER & Quality.
3. Top Command.

PartyIncome	A	A.PI<B.PI	B.	
30000	30000	<	30000 (F)	
20000			20000 (F)	= 0
25000			25000 (F)	
	20000	<	30000 (T)	
			20000 (F)	= 2
			25000 (T)	
	25000	<	30000 (T)	
			20000 (F)	= (1)
			25000 (F)	

Correlated Sub query process

inner
30000 20000
25000 = 25000

Simple Sub query

SET OPERATIONS :-

It does Operations on ~~data~~ coise

It is set theory based.

It uses various SET OPERATORS to do it function -ality.

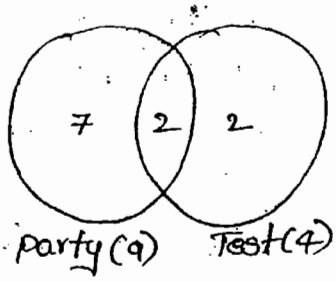
UNION (OR) UNION ALL (OR) INTERSECT (OR) EXCEPT

Syntax:-

```
SELECT COLS/* FROM TABLE A.
UNION (OR) UNION ALL (OR) INTERSECT (OR) EXCEPT
SELECT COLS/* FROM TABLE B.
```

Limitations:-

1. Numbers of Columns should be same in the both the test cases.
2. The order of the datatypes is also be same.
3. It doesn't bothers about the columnnames.



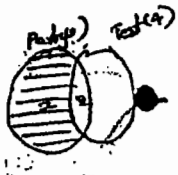
(∩) Intersect = 2

(∪) Union
(excludes duplicates) = 9+4-2 (duplicates)
= 11

(∪) UNION ALL
(includes duplicates) = 9+4 = 13

(−) EXCEPT = (party - Test)

```
SEL PARTYID, PARTYNAME FROM PARTY
EXCEPT
SEL PARTYID, PARTYNAM FROM PARTY
```



UNION:-

It merges the data from two datasets excluding duplicates.

UNION ALL:-

It merges the data from two datasets including duplicates.

INTERSECT:-

It takes common rows from both the datasets

EXCEPT:-

It takes exclusive rows from first table which are not included in the second table.

NOTE:-

performance wise do the GROUP BY operation at individual query ratherthan entire SET operations.

```
SEL PARTYID, PARTYNAME FROM PARTY
GROUP BY PARTYID, PARTYNAME
EXCEPT
SEL PARTYID, PNM FROM TEST
GROUP BY PARTYID, PNM
```

JOINS

1. It performs operations at column wise.
2. To retrieve the data from 2 or more tables we go for joins.

Types of Joins:-

- CROSS JOIN
- INNER JOIN
  - EQUI JOIN
  - SELF JOIN
  - NON EQUI JOIN
- OUTER JOIN
  - LEFT OUTER JOIN
  - RIGHT OUTER JOIN
  - FULL OUTER JOIN

COMMON JOINS SYNTAX:-

```
SEL COLS/* FROM TABLEA
CROSS JOIN TABLE B
INNER JOIN TABLE B ON CONDITION
LEFT [OUTER] JOIN TABLE B ON CONDITION
RIGHT [OUTER] JOIN TABLE B ON CONDITION
FULL [OUTER] JOIN TABLE B ON CONDITION
WHERE <RESIDUAL CONDITION>
```

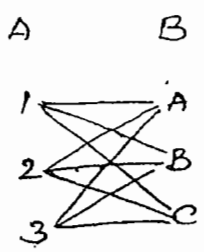
NOTE:-

This Residual condition will be applied after all the joins.

1. CROSS JOIN

1. It is the CROSS PRODUCT of Two tables.
2. If Table A contains m rows & Table B contains n rows, after CROSS JOIN you will get mxn rows.
3. There is no where condition for cross joins.

$10^{10^{100}}$ } Largest Number.



3 3 = 9 rows.

Eg:-

```
SEL P.PARTYID, P.PARTYNAME, T.ANM FROM PARTY P
CROSS JOIN TEST T.
        (or)
SEL P.PARTYID, P.PARTYNAME, T.PNM FROM PARTY P,
TEST T.
```

NOTE:-

CROSS JOIN gives the worst performance, so we need to avoid it. In REALTIME we never go for it.

2. INNER JOIN

1. It fetches the data based on the condition

a) Equi join

if the condition is based on equals operators then it is equi join.

Eg:- SEL P.PARTYID, P.PARTYNAME, T.PNM FROM PARTY P
INNER JOIN TEST T ON P.PARTYID = T.PARTYID.

Cross join = 9×9 = 36 rows
inner join(equi) = 2 rows

b) Non-Equi join:

If the condition is based on other than equals operators then it is non-equi join.

(<, <=, >, >=, <>)

Eg; SEL P.PARTYID, P.PARTYNAME, T.PNM FROM PARTY P
INNER JOIN TEST T ON P.PARTYID <> T.PARTYID.

inner join(non equi) = 9×9 - 2
= 34

Note:-
Non-Equi join some times works like cross join so we need to Avoid that in realtime by taking equals operator on condition data.

The above written Covelated query is Example for SELF JOIN.

3. OUTER JOIN

1. It Gets matched data based on the condition.
2. It Gets unmatched data from the specified Table.

Note:- In case of unmatched the other table data will be Null.

RIGHT OUTER JOIN;

1. It gets matched data based on the condition.
2. It gets unmatched data from the left table.

Eg: Sel p.partyid, p.partyname, T.Testid from party p
Left join Test T on p.partyid=T.partyid

Note:- In case of unmatched the Right Table data will be null.

RIGHT OUTER JOIN;-

1. It gets matched data based on the condition.
2. It gets unmatched data from the Right Table.

Eg: Sel p.partyid, p.partyname, T.Testid from party p
Right join Test T on p.partyid=T.Partyid.

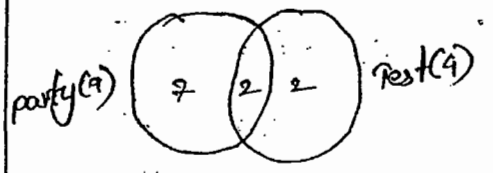
NOTE: In case of unmatched the left Table data will be null.

FULL OUTER JOIN;-

1. It gets matched data based on the condition.
2. It gets unmatched data from the Left & Right table.

Eg: SEL P.PARTYID, P.PARTYNAME, T.TESTID FROM PARTY P,
FULL JOIN TEST T ON P.PARTYID = T.PARTYID.

NOTE: IN CASE OF unmatched the other table data will be null



Cross join = 9×4 = 36

inner join = 2 rows.

inner join (non-equi) = 9×4 − 2
<> = 34

Left join = 2+7 = 9 rows

Right join = 2+2 = 4

Full join = 2+7+2 = 11 rows.

O/P :-

	PARTY		TEST
	PARTYID	PARTYNAME	TESTID
	2	koosh	0001
	3	Vinay	0002
Left join —	4	madhu	(?) — null
null —	(?)	?	0004 — Right join

Important Questions:-

1. Difference b/w Subquery & Correlated Subquery?
2. Difference b/w Orion & Orionall
3. Display top 4 Employee Salaries.
4. Difference b/w cross join & full outer join.
5. CUSTOMER TABLE

CID	CNAME
1	ABC
2	MNO
3	RTV
4	DEL

CALLS TABLE

CID	CALLID
1	0001
2	0002

A) Display all customers who are available in customer table and not available in calls Table?

NOTE: Do with Only Joins, Don't use NOT IN, <>, NOT EXISTS OPERATORS.

```
SEL CT.CID, CT.NAME, CA.CALLID FROM FROM CUSTO
MER CT LEFT JOIN CALLS CA ON CT.CID = CA.CID
WHERE CA.CALLD IS NULL.
```

CID	CNAME	CALLID
1	ABC	0001
2	MNO	0002
3	RTV	(9)
4	DEL	(9)

VIEWS

It is a window for the Table.

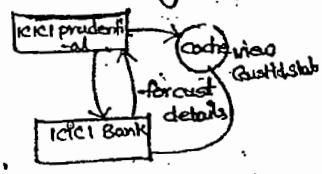
It is the logical object for the physical collection of Table.

The views only gives mainly for restricting only rows and no. of columns.

B) Reducing network bandwidth.

c) providing better encapsulation.

d) " " security



NOTE:- In REALTIME in two situations we used VIEWS

a) Restricting Rows and columns.

b) Creating Datamarts

Based on functionality these are 3 types of VIEWS

A) READ ONLY

B) UPDATEBLE

c) MATERIALIZED.

SYNTAX:- Creating:-

CREATE VIEW <VIEWNAME> AS (SELECT QUERY <where condition> <with check option>)

Replacing:-

REPLACE VIEW <VIEWNAME> AS (SELECT QUERY <where condition> <with check option>>)

Dropping :-

DROP VIEW <VIEWNAME>

CALLING a View :-

SEL * FROM <VIEWNAME>

Updating a View :-

UPDATE <VIEWNAME> SET COLNAME=<VALUE> <WHERE CONDITION>

2) UPDATABLE VIEW :-

If you create a view based on a single table then it will be updatable view. Here we can perform Insert, Update and delete operations on views so that table will be effected.

CREATE VIEW vw_party2 AS (SEL * FROM party)

SHOW VIEW vw_party2

SEL * FROM vw_party2

REPLACE VIEW vw_party2 AS (SEL partyid, partyname, partyincome FROM party)

UPDATE vw_party2 set partyincome = 77777 WHERE partyid = 1.

DONLY VIEW; (JOIN VIEW & AGGREGATE VIEW):-

Here we can't perform any operations.

JOIN VIEWS;

Creating a view based on more than one table.

CREATE VIEW vw_party3 AS(
SEL p.partyid, t.partyname from party p
LEFT OUTER JOIN test t on p.partyid=t.partyid);

SHOW VIEW vw_party3

SELECT * FROM vw_party3;

REPLACE VIEW vw_party3 AS(SEL t.partyname from party p LEFT OUTER JOIN test t ON p.partyid = t.partyid);

AGGREGATE VIEWS;

Aggregate views are views which contain aggregate expressions or derived columns. It is always necessary to assign a name to these columns. so that they may be referenced with in the views.

An optional LOCKING ..... FOR ACCESS CLAUSE may be added to the query which permits the table rows to be read even in the event that another read or write lock is held on the table.

Eg:-

```
CREATE VIEW partysals AS
LOCKING TABLE party FOR ACCESS
SEL partycode AS department,
SUM(partyincome) AS salary_total,
AVG (partyincome) AS salary_Average,
MAX (partyincome) AS salary_Maximum,
MIN (partyincome) AS salary_Minimum.
 FROM party
GROOP BY partycode.
```

*** * VIEWS WITH CHECK OPTION:-

The WITH CHECK option prevents rows from being inserted or updated via the view, if the resulting rows fall outside the view.

```
CREATE VIEW dept_budget as
SEL partyid, partyincome FROM party
WHERE partyincome <=100000 WITH CHECK OPTION;

SEL * FROM party

INSERT INTO dept_budget(partyid, partyincome)
 VALUES (603,700000);
```

* Failure 3564 Range Constraint: check error in field party.partyincome.

Department partyIncome, inserted or updated via this view, may not exceed $1,00,000.

NOTE:-

CHECK constraint can also apply at a table level creation & Views also.

MACROS	PROCEDURES
1. It is collection of statements which does simple tasks.	1. It is precompiled collection of statements which does Complex task Ex:- Error handling, Working with DBX operations etc;
2. Macro execution results are catched so, that it gives fast response	2. Procedures always runs at the Server Side.
3. For frequent & simple request we go for it.	3. For Rare request & Complex operations.
4. All Stmts inside the macro body executed like a transaction(if one stmt failed then macro failed)	4. Not Applicable.
5. We Cannot take procedural Control Stmts (if, if else, for etc;) inside MACRO body.	5. Procedure we can take.
6. Max 1 DBX it Supports	6. No restriction like that.
7. Takes only i/p parameters	7. procedures A) IN: i/p Value to the proced B) OUT: " from " C) INOUT: Takes i/p value & pa-es o/p value from th proced

8. Security is less.	8. Security is More.
9. Macro executed by using EXEC (or) EXECUTE STMT.	9. Executed by using CALL statement.

MACROS:-

Creating:

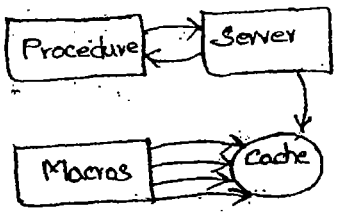
CREATE MACRO <MACRONAME>(<PARAMS>) AS (SQL QUERIES;);

Replacing:

REPCACE MACRO <MACRONAME> (<PARAMS>) AS (SQL QUERIES;);

Dropping:

DROP MACRO <MACRONAME>



Viewing:-

SHOW MACRO <MACRONAME>

Executing a Macro:-

EXEC <MACRONAME>(<PARAMS>) OR EXECUTE <MACRONAME>(<PARAMS>)

CREATE macro mc_party2(partyid1 integer) AS (SEL * FROM party where partyid=:partyid1;)

SHOW MACRO mc_party2

EXEC mc_party2(2)

```
REPLACE MACRO mc_party2(partyid1 integer,
partyincome1 integer) AS (SEL * FROM party1d =:
partyid1 or partyincome >: partyincome1;)
EXEC mc_party2(2,20000)
```

5.11 USING MACROS TO PRESERVE INTEGRITY:-

Macros may be used for either of the following purposes:

1) To insure data integrity - that the data conforms to certain range or value restrictions.
2) To insure referential integrity - that the data being input as foreign keys have legal primary key values elsewhere in the database.

Create a macro which inserts a new party member row with the following restrictions:

1) party must be over 21 years of age.
2) party must be assigned a valid partycode.

```
CREATE MACRO new_party
(pid INTEGER, pname VARCHAR(30), pcode INTEGER,
hired DATE, birth DATE)
AS
ROLLBACK WORK 'Invalid Hire'
WHERE (:hired - :birth)/365 <21;
ROLLBACK WORK 'Invalid Party Code'
```

WHERE :pcode NOT IN (SELECT partycode FROM party WHERE partycode=:pcode);

INSERT INTO party(partyid, partyname, partycode, jdate) VALUES(:pid, :pname, :pcode, :hired)););

fail: exec neco_party(19,'meer',40,'2010-08-06','2010-08-06')

Success: exec neco_party(19,'meer',40,'2010-08-06','2981-08-06')

PROCEDURES:- Creating:-

CREATE PROCEDORE <PROCEDORENAME> (<PARAMS>) AS BEGIN <SQL QUERIES> END

Replacing:-

REPLACE PROCEDORE <PROCEDORENAME>(<PARAMS>) AS BEGIN <SQL QUERIES> END.

Dropping:-

DROP PROCEDORE <PROCEDORENAME>

Viewing:-

~~VIEW~~ SHOW PROCEDORE <PROCEDORENAME>

Calling:-

CALL PROCEDORE <PROCEDORENAME>(<PARAMS>)

EG1:

```
CREATE PROCEDURE P_SAMPLEP(DCODE1 INTEGER,
DNAME1 VARCHAR(30))
BEGIN
DECLARE DEPTNAME VARCHAR(30);
SET DEPTNAME=DNAME1;
INSERT INTO DEPT(DCODE,DNAME) VALUES(:DCODE1,:
DEPTNAME);
END;

CALL P_SAMPLEP(70,'SEVENTY');

SELECT * FROM DEPT;
```

EG2:

```
CREATE PROCEDURE WITHDRAWAL
(ACCTID INTEGER, AMOUNT DECIMAL(9,2))
 BEGIN
    DECLARE CURBAL DECIMAL(9,2);
    SEL BALANCE INTO :CURBAL
    FROM ACCOUNTS WHERE ACCTID=:ACCTID;
    IF AMOUNT <=CURBAL THEN
        UPDATE ACCOUNTS SET BALANCE=BALANCE-:
  AMOUNT WHERE ACCTID=:ACCTID;
        INSERT TRANSLOG VALUES(CURRENT_TIMESTA
  :ACCTID,'W',:ACCOUNT);
```

This table gives history into

END IF;

END;

TRIGGERS

It does an action when event occured.

Generally to prevent unauthorized access on tables we go for triggers.

while doing insert, update & delete operations on table for allowing or not allowing the operations and doing some actions we go for.

Based on the event there are two types of triggers

a) Before trigger event: there action first event next.

b) After trigger event; there event first and action next.

There we apply trigger operations at two levels.

a) for each row level

(update <tablename> set colname = value where <condition>

b) for each stmt level.

(update <TN> set colname = value)

SYNTAX:-

CREATE TRIGGER <TRIGGERNAME>( )

REPLACE TRIGGER <TRIGGERNAME>( )

DROP TRIGGER <TRIGGERNAME>

HELP TRIGGER <TRIGGERNAME>

SHOW TRIGGER <TRIGGERNAME>

ALTER TRIGGER <TRIGGERNAME> DISABLED.

ALTER TRIGGER <TRIGGERNAME> ENABLED.

CREATE TRIGGER <TRIGGERNAME> AFTER UPDATE OF (COLUMNNAME) ON <TABLENAME> REFERENCING OLD TABLE AS OLD TABLE NEW TABLE AS NEW TABLE

FOR EACH STATEMENT/ROW

[WHERE CONDITION]

(TRIGGER ACTION)

CREATE TRIGGER VINAY AFTER UPDATE OF (PARTYIN COME) ON PARTY

FOR EACH STATEMENT

(INSERT INTO TEST(11, 'Record updated', '00001'));

SHOW TRIGGER VINAY

UPDATE PARTY SET PARTYINCOME=9.9999' WHERE PARTYID=2

The following statements are not allowed inside a stored procedure called from a trigger:

→ DDL statements

→ DCL sta

→ BT (BEGIN TRANSACTION) -- ET (END TRANSACTION)

→ COMMIT

→ EXCEPTION HANDLING STATEMENTS.

→ IN OUT and OUT parameters are not allowed in a stored procedure called from a trigger.

→ A row can be passed to a stored procedure, but a table cannot.

NOTE:-

OLD:- Refers to the old table values before event.

NEW:- The values After trigger raise.

PARTYID	PARTYNAME
2	10000

old.pincome: 10000

new.pincome: 99699

CURSORS

Definition:-

A cursor is a data structure that stored procedures and preprocessors use at runtime to point to the result rows in a response set returned by an SQL query.

Types of Cursors:-

You can declare the following types of cursors in a DECLARE CURSOR statement:

- Dynamic
- Macro
- Request
- Selection
- Stored procedure.

Stored Procedures only support stored procedure-type cursors.

The following types refer two ways you can use a cursor to manipulate data:

- positioned (updatable)
- Non-positioned (read only)

These are two types of Cursors:

A. FOR LOOP CURSOR:

1. The scope of the cursor confined to the FOR LOOP only.

B. DECLARE CURSOR:

1. The scope of the cursor confined to the Begin and End statements.

CURSOR PROCESS: (5 STEP PROCESS):

Following is the general process flow for updating or deleting a row using a DECLARE cursor cursor in a stored procedure:

1. Specify a DECLARE CURSOR statement with the appropriate cursor specification.
2. Open the cursor with an OPEN cursor_name statement.
3. Execute FETCH statements to fetch one row at a time from the response set. The next cursor movement depends on the scrollability option you specify.
4. Update or delete a fetched row using the WHERE CURRENT OF clause with an UPDATE or DELETE statement, respectively.
5. close the cursor by performing a CLOSE cursor_name statement.

If cursor_name is not open at the time CLOSE is submitted.

- SQLCODE is set to 7631
- SQLSTATE is set to '24501'

If there are no rows in the response set at the time you execute FETCH

- SQLCODE is set to 7362
- SQLSTATE is set to '02000'

EG(DECLARE CURSOR):

```
CREATE PROCEDURE sp3C()
    BEGIN
        DECLARE var1 INTEGER;
        DECLARE var2 CHARACTER(30);
        DECLARE C1 CURSOR FOR
        SELECT partyid, partyname FROM party
        ORDER BY partyid;
        OPEN C1;
        WHILE(SQLCODE=0) DO
        FETCH C1 INTO VAR1, VAR2;
        INSERT INTO PARTY1(PARTYID, PARTYNAME)
        VALUES(var1, var2);
        END WHILE;
        CLOSE C1;
    END

CALL sp3C()
Select * from party1;
```

TERADATA UTILITIES

The Utility nothing but a tool / application (or) collection of stmts as a script which loads and unloads the data.

There are nearly 45 utilities in Teradata out of them they very important are.

OPERATIONS		BETWEEN
BTEQ	: LOADING/UNLOADING	[FILE→TABLE, TABLE→FILE]
FAST LOAD	: LOADING	[FILE→TABLE]
MULTILOAD	: LOADING	[FILE→TABLE]
TPUMP	: LOADING	[FILE→TABLE]
FASTEXPORT	: UNLOADING	[TABLE→FILE]
OLELOAD	: LOADING	[DB/FILE↔FILE/DB]
TERADATA PARALLEL TRANSPORTER	: LOAD	
TERADATA PARALLEL TRANSPORTER API	: LOAD	
QUERYMAN	: LOADING/UNLOADING	

Queryman → File → Import Data

UTILITY MODES

INTERACTIVE MODE
[One by one statement we fire here]

BATCH MODE
[Collection of stmts as a script we fire here]

TERADATA BATCH MODE:-

1. We can write set of sql stmts. inside (any editor) (notepad, wordpad etc;) and save with any extension such as .txt, .tat, .out etc;
2. To differentiate one script to another script in teradata it recomended BTEQ → .btq
   Fastload → .fld
   Multiload → .mld
   Tpump → .tmp
   Fastexport → .fxp etc;

Executing Utilities:-

1ST WAY:-

(*** BY USING THIS WE CAN RUN A SCRIPT INSIDE ANOTHER SCRIPT)

.RUN FILE =<FILENAME>

2ND WAY:-

IN WINDOWS:

BTEQ: START → RUN → CMD → BTEQ < [SCRIPT PATH] > [LOGFILEPATH]

FLOAD: START → RUN → CMD → FASTLOAD < [SCRIPT PATH] > [LOGFILEPATH]

MLOAD: START → RUN → CMD → MLOAD < [SCRIPT PATH] > [LOGFILEPATH]

TPUMP: START → RUN → CMD → TPUMP < [SCRIPT PATH] > [LOGFILEPATH]

FEXP : START → RUN → CMD → FEXP < [SCRIPT PATH] > [LOGFILEPATH]

IN UNIX:-

BTEQ: BTEQ < [SCRIPT PATH] | TEE <LOGFILEPATH>

FLOAD: FASTLOAD < [SCRIPTPATH] | TEE <LOGFILEPATH>

MLOAD: MLOAD < [SCRIPT PATH] | TEE <LOGFILEPATH>

TPUMP: TPUMP < [SCRIPT PATH] | TEE <LOGFILEPATH>

FEXP : FEXP < [SCRIPT PATH] | TEE <LOGFILEPATH>

NOTE: TEE displays the LOG INFORMATION ON THE CONSOLE as well as REDIRECTS it to the specified file.

RETURN CODES AND SYMBOLS:-

0 — job completed normally

4 — warning

8 — user error

12 — fatal error (connection Related issues come, here)

16 — if no msg destination is available or typical error:

- In Unix to know the script ran Success or not:

if [$?] = 0 means script success

if [$?] = other than 0 means script fail.

GENRAL Utilities Commands & Keywords:-

Almost all Utilities and starts with dot(.).

Normally sql stmts end with (;)

Comment Notation

/* */ BTET MODE

-- ANSI MODE

BTEQ:-

A) • ERRORCODE: It returns 0 if the previous sql stmt is succeded.

B). ACTIVITYCOUNT: It returns other than 0 if the previous sql stmt is failed.-

C) • ERRORLEVEL: It sets the Sevierity level for the error code at error handling.

D) • REMARK: It displays message on the console.

Ex:- .Remark 'Vinay here' O/P: Vinay here

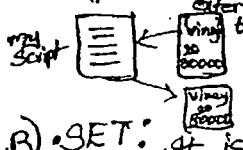
E) • REPEAT: It repeats the below stmts to the specified numbers of times.

Ex:- • Repeat *; // till last record in the i/p file it repeats

• Repeat 2; // only 2 times it repeats.

ALL UTILITIES:

A) • ACCEPT: It takes external value from a file. It reads only 1 record from external file.



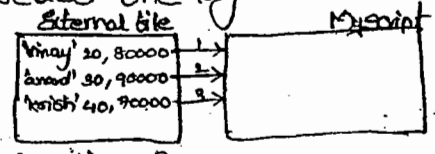
B) • SET: It is used to set the internal variable. in the

Script.

C). DISPLAY: It displays the msg in the specified file.

D). ROOTE MESSAGES: It redirects the log information to the specified file.

E). IMPORT: It reads one by one record from a exter-nal file.



F). EXIT: It quits the screen.

G). QUIT: It quits the window & comes out of the loop.

H). OS CLS: It clears the screen.

I). SHOW CONTROLS: It displays all environmental controls or the current settings in BTEQ.

J). QUIET ON/OFF: It limits the no. of error and request crossing statements.

K). LOGON: Logging into the utility.

L). LOGOFF: Logout from the utility.

Additional Commands Without .(DOT):-

1. TENACITY: The no. of hours it takes to establish a connection to the servers.

2. SLEEP: It sleeps the specified no. of hours minutes between multiple logon requests.

3. CHECK POINT: It is used for restarting purpose.

4. SESSIONS: It is a commit interval between various records loading into the Database.

5. PACK: No. of statements/request.

6. RECORD: Reads the specified no. of records.
   Syntax: Record <startvalue> THRU <Endvalue>
   Eg:- Record. 1 THRU 10

7. USING: It defines the structure of i/p file in BTEQ.

8. DEFINE: Defines i/p file structure in fastload.

9. LAYOUT: Defines i/p file structure in MULTILOAD & TPUMP.

RUNNING A SCRIPT FROM ANOTHER SCRIPT:

- RUN FILE = <FILENAME>

B). ACTIVITYCOUNT:-

It displays the no. of rows returned in the previous sql statement.

Ex:-

1) select * from party Party 14 rows

ActivityCount=14

2) Select Count(*) from party;

ActivityCount=1.

4. SESSIONS;

Increasing the no. of sessions we also increas the parallelism

UTILITIES LOGIN:

INTERACTIVE;

SY:.LOGON IPADDRESS OR SERVERNAME/<USERNAME> <ENTER KEY> PASSWORD=<PASSWORD>

Eg:-

TD12:→.LOGON 127.0.0.1/DBC <ENTERKEY> PASSWORD=DBC

BATCH MODE;-

SY:.LOGON IPADDRESS OR SERVERNAME/<USERNAME>,<PASSWORD>

Eg:-

TD12:→.LOGON 127.0.0.1/DBC,DBC

BTEQ

[Basic TeraData Query]

Runs in INTERACTIVE MODE

Runs in BATCH MODE

Navigation:-

a) Start → run → BTEQ

b) start → All programs → Teradata client → Teradata BTEQ.

To See all environmental Controls then we go for

- SHOW CONTROLS

Setting some of the Commands.

PID	PARTYNAME
1	X
2	Y
3	Z

- SET SIDETILES ON

SEL * FROM PARTY

PID 1 PARTYNAME X
PID 2 PARTYNAME Y
PID 3 PARTYNAME Z

- SET SIDETILES OFF

NE ON

SEL * FROM PARTY

PID	PNAME
1	x
2	y
3	z

*SET FOCDLINE OFF

**These Commands are called Report Commands

TERADATA TRANSACTION SEMANTICS

BTET	ANSI
[Begin Transaction End Transaction OR BASIC TERADATA TRANSACTION]	[American National Standard Institute]
→ Default mode	→ Explicit Mode
→ Auto Commit	→ Need to Commit & rollback explicitly
→ Default table multiset	→ Default multiset.
→ Mainly for teradata related queries & Commands	**→ Any ANSI Complaint database queries & Commands it executes [oracle, Sqlserver---]

Changing Modes from BTET to ANSI:-

Before LOGON you do this

- SET SESSION TRANSACTION ANSI ↵

Then Now LOGON

- LOGON 127.0.0.1/DBC ↵

Password: *** ↵

REALTIME USAGE OF INTERACTIVE MODE:-

A) However we are working with queryman the same will be work with BTEQ for giving Normal queries, doing normal operations.

Rules while Exporting the data:-

a) Filepath, structure, Format

b) select statement to retrieve the data.

Rules for importing the data:-

a) Filepath, structure, Format

b) insert statement to load the data.

BTEQ BATCH MODE UTILITY:-

1) It imports & exports the data.
2) For giving the queries generally and working with many DML operations (or) Creating stage table we go for it.
3) Mainly used for
   a) Importing, Exporting from multiple tables to multiple files.
   b) Doing ETL operations. (Extract, Transform, Load)
   c) Doing Exception handling
   d) Implementing Branching & looping (Control flow)
4) It doesn't bothers, that the table is having contains constraints, indexes, triggers etc;
5) It loads duplicates also.
6) It performs insert, update, delete etc, & other operations also.
7) It is not automatic restartable & checkpoint configurable.
8) It doesn't runs in seperate steps.
9) It does exception handling but not error capturing.

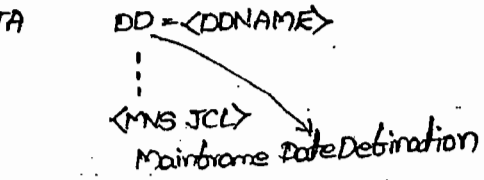
Note:- fastload only we can perform insert operation only.

10) BTEQ Contains report formatting Commands as well.

EXPORTING OF DATA:-

If you want export the data then go for

- EXPORT DATA "FILE=<FILENAME>", LIMIT N
  INDICDATA DD=<DDNAME>
  REPORT
  DIFF
  RESET



IMPORTING OF DATA:-

- IMPORT DATA "FILE=<FILENAME>", SKIP N
  INDICDATA DD=<DDNAME>
  REPORT
  VARTEXT 'D'

DATA MODE:-

It is record by record in format mode. Here the Nulls are represented with 0's and spaces.

DATA	DATA
DATA	DATA

INDICDATA MODE:-

It is just like DATA Mode but NULLS are indicated with indicator bytes.

INDIC DATA	DATA	DATA
INDIC DATA	DATA	DATA

data is in DATA or INDIC data Mode we cannot read the numerical info.

REPORT MODE:-

Here field by field info. will be Exported or imported

PARTYID	PARTYNAME
1	A
2	B

DIFF MODE:-

Data interchange in file format:-

It is the format used across various PC based platforms. such as LOTUS 1-2-3.

VARTEXT :-

It is the Variable Text format Mode used frequently in the realtime. Generally denoted with delimeters such as TAB, | etc;

FILE:-

FILENAME in the N/w Attached System

DD - Data Definition in the channel Attached System.

LIMIT:-

Limiting the No. of rows while exporting the data.

SKIP:-

Skipping the No. of rows while importing the data.

RESET:-

It reverses the effect of previous EXPORT command

and closes the output and the console.

Eg 1:- Exposting the Data in the DATA Mode.

go to Notepad

```
.LOGON 127.0.0.1/DBC,DBC
 DATABASE VINAYAKA;
.EXPORT DATA FILE=C:\DATA_LOG\PARTY1_DATA.TXT
 SELECT * FROM PARTY;
.EXPORT RESET
.LOGOFF
```

Then Save with EXPORT_DATA.txt in C:\BTEQ\ folder.

The Run Script.

Start → RUN → CMD → BTEQ<C:\BTEQ\EXPORT_DATA.txt >C:\DATA_LOG\EXPORTLOG.txt

(optional.)

Eg2:- Exporting the Data in the REPORT Mode the following business logic.

ⓐ Employee Address should be the combination of Name & Location concatination.

ⓑ ~~Partyincome should be~~ partyrating should be according to the below process.

Go to Notepad.

```
.LOGON 127.0.0.1/DBC,DBC;
.SET WIDTH 600;
DATABASE VINAYAKA;
.EXPORT REPORT FILE=C:\DATA_LOG\PARTY1_REPORT.TXT
SEL PARTYID AS PARTYREPID, CAST(PARTYNAME ||'_'||
 PARTYLOC AS VARCHAR(30)) AS PARTYADDRESS,
 PARTYINCOME,
 CASE
     WHEN PARTYINCOME < 30000 THEN 'POOR'
     WHEN PARTYINCOME > 30000 AND PARTYINCOME <
      30000 THEN 'AVERAGE'
      ELSE 'GOOD'
 END AS PARTYRATING,
 JDATE(FORMAT 'DD-MM-YYYY') AS JDATE FROM
 PARTY;
 .EXPORT RESET
 .LOGOFF
```

FREQUENTLY USED FORMAT'S WHILEE EXPORTING & IMPORTING :-

Fixed Width Format

```
123 VINAY
456 MADHO
789 KRISH
```

30% Use in Realtime

Delimited format (VARTEXT Format)

```
123 VINAY
3456 MADHO:KUMAR
8 VISHNU
```

delimeter

70% use in realtime.

Exporting FixedWidth Data :-

goto Notepad

```
•LOGON 127.0.0.1/DBC,DBC;
DATABASE VINAYAKA;
•EXPORT REPORT FILE=C:\DATA_LOG\PARTY1_FIXEDWIDTH.TXT
SELECT CAST(PARTYID AS CHAR(2)) || CAST(PARTYNAME
AS CHAR(5)) || CAST(PARTYINCOME AS CHAR(5)) AS
TEXT_DESC FROM PARTY;
•EXPORT RESET
•LOGOFF
```

Save with EXPORT_FIXEDWIDTH.

Exporting Variable Fixed. Data ;-

In the above Script do the below change.

```
SELECT CAST(PARTYID AS VARCHAR(5)) || ',' ||
  CAST(PARTYNAME AS VARCHAR(10)) || ',' ||
  CAST(PARTYINCOME AS VARCHAR(10)) AS TEST_DESC
  FROM PARTY;
```

Importing the Data in DATA MODE:-

```
•LOGON 127.0.0.1/dbc,dbc;
•import data file='C:\DATA_LOG\PARTY1_DATA.TXT';
 DATABASE VINAYAKA;
 •quiet on; → Error msgs & requests stmts will take
 •repeat *;
  USING i_partyid(INTEGER),
```

```
    i_partyname(varchar(30))
INSERT INTO PARTY1(
        partyid, partyname)
 VALUES (:i_partyid, :i_partyname);
.LOGOFF
```

NOTE:- The input file should be data mode to import the data.

Importing Variable text data Mode & Doing ETL Operations:-

```
.logon 127.0.0.1/DBC,DBC;
.import vartext ',' file="C:\BTEQ\DATA_VARTEXT1.TXT",
 SKIP=1;
DATABASE VINAYAKA;
quiet on;
.repeat *;
USING
      i_org(varchar(20)),
      i_cust_nbr(varchar(5)),
      i_partyname(varchar(30)),
      i_partyhname(varchar(10));
INSERT INTO party_tgt(partyid,partyemail,jdate,
partyincome) VALUES(:i_org||'_'||:i_cust_nbr,
teradatavinay@yahoo.co.in, CORRENT_DATE,
:i_partyincome*12/100);
.LOGOFF
```

DATA_VARTEXT1.txt:- (Input file)

ORD, CODE, NAME, INCOME
IBM, 100, MAHESH, 60000
TCS, 200, KISHORE, 50000
WIPRO, 300, RAJESH, 400000

Implementing Branching & Looping:-

Eg:-

```
.LOGON 127.0.0.1/dbc,dbc;
 database vinayaka;
 select * from party;
 .if errorcode=0 then .goto vinay1bl
 .Remark 'NO TABLE EXISTED'

 .label vinay1bl
 .if Activitycount >5 then .Remark 'TABLE EXISTED
  AND CONTAINED MORE THAN 5 ROWS'
 .LOGOFF.
```

Error Handling:-

To prevent from known and unknown errors at runtime and doing our user defined Action is called error handling.

→ we need to set errorlevel for Handling the Errors.

Syntax: .SET ERRORLEVEL <ERRORCODE> SEVERITY <INTEGER>

Eg:- .SET ERRORCEVEL 6784 SEVERITY 4

.SET ERRORLEVEL 3807 SEVERITY 8

.SET ERRORLEVEL 2121 SEVERITY 12

.SET ERRORLEVEL 5698 (OR) ONKNOWN SEVERITY 16

NOTE:- HIGHEST RETORN CODE=0 MEANS SCRIPT SUCCESS.

Eg:- ERRORHANDLING.TXT

```
.logon 127.0.0.1/dbc,dbc;
 database VINAYAKA;
.set errorlevel 3807 Severity 4
.set errorlevel 6066 Severity 8
 select * from party22;
.if errorlevel=4 then .remark 'client didnt created
      please dont load the table';
.if errorlevel=8 then .remark 'range invalid, don't
      load the table';
.LOGOFF.
```

FAQ's:-

1. Difference b/w Errorcode & Errorlevel.
2. What is the main imp. of BTEQ in realtime.
3. There is a file it contain 100 records, load 60 records by skipping first 20 & last 20 records.

```
.SKIP=20          20
.Repeat 60       ↓60
                 20
```

FASTLOAD UTILITY :-

1) It loads data into empty table.
2) It loads only one table at a time.
3) Compare to all utilities it is faster.
4) It cannot load duplicate Records [either multiset or NUPI(column level)]
5) It loads error data in to error tables.
6) It is fully automatic restartable & checkpt configurable
7) The table shouldn't have indexes, Constraints etc
8) It does only insert operation b/w BEGIN & END LOADING statements.

100000 records → Table.
File

1-100000 reads first → 1 lakh records load DB.

1-100000 90,000 failed DB

when Restart.

It loads from again first then we go to check point configurable.

1-10000 → 10000
10000-20000 → 20000
:
90000-1lakh → Failed.

Restart it 90001 it loads.

9) It supports automatic transient generally.

10) It does automatic data conversion b/w
   A) Numeric to Character.
   B) ~~the~~ Date to character
   C) Character to Date.
   D) Character to Numeric.

11) It runs in two phases. [BTEQ runs 5 phases]

12) Maximum 15 fastload scripts you execute at a time

13) Generally in realtime for loading into stagetables or company's initial loading we go for it.

14) It supports i/p module programming (INMOD) for its operations.

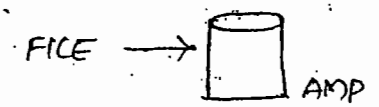
NAVIGATION:-

A) Start → RUN → FASTLOAD.

B) Start → PROGRAMS → TD CLIENT → TD FASTLOAD.

FASTLOAD PHASES:-

PHASE 1 - ACQUISITION / INSERT PHASE.



- Getting the data
- unordered data

- PE loads blocks of data to each AMP.
- AMPs hash each record and redistribute to appropriate AMP.
- Records written to target table on unsorted blocks of data.

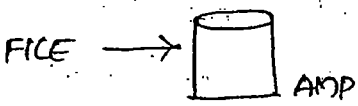
9) It supports automatic transient generally.
10) It does automatic data conversion b/w
   A) Numeric to Character
   B) the Date to character
   C) Character to Date.
   D) character to Numeric.
11) It runs in two phases. [BTEQ runs 5 phases]
12) Maximum 15 fastload scripts you execute at a time
13) Generally in realtime for loading into stagetable or company's initial loading we go for it.
14) It supports i/p module programming (INMOD) for its operations.

NAVIGATION:-

A) Start → RUN → FASTLOAD.
B) Start → PROGRAMS → TD CLIENT → TD FASTLOAD.

FASTLOAD PHASES:-

PHASE 1 - ACQUISITION / INSERT PHASE.



- Getting the data
- unordered data

- PE loads blocks of data to each AMP.
- AMPs hash each record and redistribute to appropriate AMP.
- Records written to target table as unsorted blocks of data.

FASTLOAD UTILITY :-

1) It loads data into empty table.
2) It loads only one table at a time.
3) Compare to all utilities it is faster.
4) It cannot load duplicate Records [Either multiset or NUPI(Column level)]
5) It loads error data in to error tables.
6) It is fully automatic restartable & checkpt configurable
7) The table shouldn't have indexes, Constraints etc;
8) It does only insert operation b/w BEGIN & END LOADING statements.

100000 records → Table.
File

1-100000 reads first → 1lakh records load DB.

1-100000 90,000 failed DB.

when Restart.

It loads from again first then we go to check point configurable.

1-10000 → 10000

10000-20000 → 20000

⋮

90000 - 1lakh → Failed.

Restart it 90001 it loads.

FASTLOAD UTILITY :-

1) It loads data into empty table.
2) It loads only one table at a time.
3) Compare to all utilities it is faster.
4) It cannot load duplicate Records [Either multiset or NUPI(Column level)]
5) It loads error data in to error tables.
6) It is fully automatic restartable & checkpt configurable
7)

100000 records → Table.
File

1-100000 reads first → 1 lakh records load DB.

1-100000 90,000 failed DB

when Restart.

It loads from again first then we go to check point configurable.

1-10000 → 10000

10000-20000 → 20000

⋮

90000 - 1lakh → Failed.

Restart it 90001 it loads.

7) The table shouldn't have indexes, Constraints etc;
8) It does only insert operation b/w BEGIN & END LOADING statements.

10) It supports automatic transient generally.

11) It does automatic data conversion b/w
   A) Numeric to Characters.
   B) the Date to Character
   c) Character to Date.
   D) Character to Numeric.

11) It runs in two phases. [BTEQ runs 5 phases]

12) Maximum 15 fastload scripts you execute at a time

13) Generally in realtime for loading into stagetables or company's initial loading we go for it.

14) It supports i/p module programming (INMOD) for its operations.

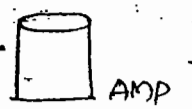
NAVIGATION:-

A) Start → RUN → FASTLOAD.

B) Start → PROGRAMS → TD CLIENT → TD FASTLOAD.

FASTLOAD PHASES:-

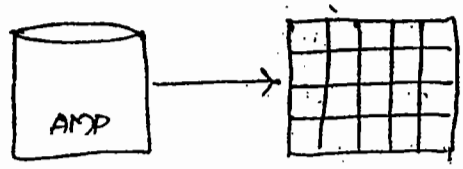
PHASE 1 - ACQUISITION / INSERT PHASE:



- Getting the data
- unordered data

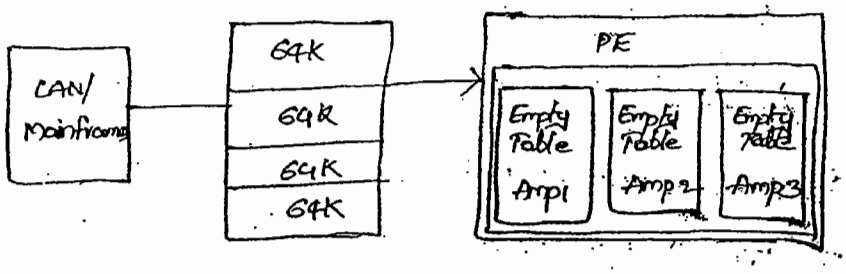
- PE loads blocks of data to each AMP.
- AMPs hash each record and redistribute to appropriate AMPs.
- Records written to target table on unsorted blocks of data.

PHASE 2 : APPLICATION PHASE [END LOADING PHASE]



- At completion of data loading, each AMP sorts target table, put rows into blocks and writes blocks to disk.
- Fallback rows are generated if FALLBACK specified

BLOCK BY BLOCK OPERATIONS :- [FASTLOAD]



1) It process the data block by block.

FASTLOAD LIMITATIONS :-

1) It doesn't support duplicate records. if table is the multiset and you're trying to store duplicate data it doesn't support duplicate loading but shows no. of duplicates in the LOG File.
2) If the table contains SI, RI, JI, HI & Triggers it cannot load the data.

ENDED PROCESS IN REALTIME:-

① DROP/DISABLE CONSTRAINTS/INDEXES./TRIGGERS.

② FASTLOAD/MULTILOAD SCRIPTS RUN.

③ CREATE/ENABLE CONSTRAINTS/INDEXES/TRIGGERS.

~~ERROR~~

LIMITATION:-

It loads empty table & performs only insert operation into the single table.

ERROR TABLES:-

They are 2 types

If don't create the system creates automatically with below names.

Ⓐ ET_ <TABLENAME>

Ⓑ UV - <TABLENAME>

1) ERRORTABLE 1 (ET TABLE):-

Generally the below types of the errors moves into the table.

a) data conversion

b) Const. Violation.

c) Unavailable Amp error.

structure:-

DBC.ERRORMSGS.

File which contains the input data bad.

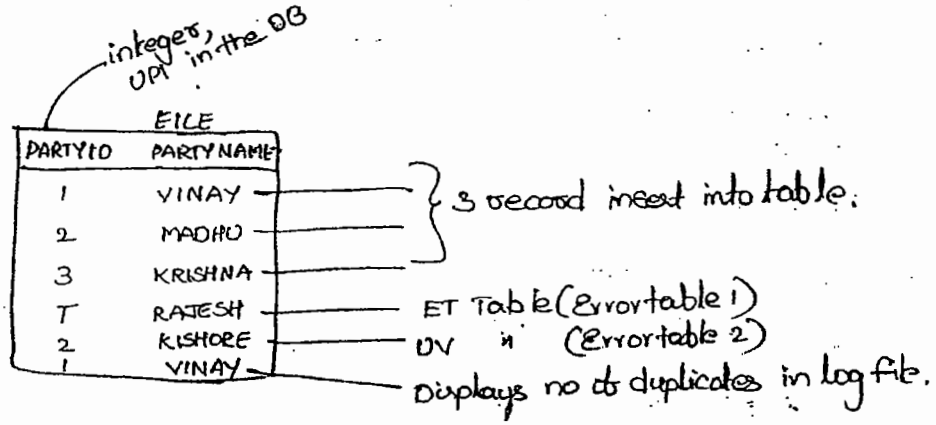
ERRORCODE	ERRORFIELD	DATAPARCEL
2378	PARTYID.	000001.DAT

2) ERRORTABLE2 (UV TABLE) [UV - UNIQUE VIOLATION]

If table is UPI and your trying to store duplicate data then those records moved into UV Table.

STRUCTURE; Similar to target table.

FASTLOAD FORMULA:-



No. of records in source file (6) =

No. of record in Target (3)

+

No. of record in ErrorTable1 (1)

+

No. of record in ErrorTable2 (1)

+

No. of duplicates in log file (1)

= 6

FASTLOAD STRUCTURE :-

```
• SET RECORD <RECORDS FORMAT>
• LOGON
BEGIN LOADING <DATABASENAME.TABLENAME>--PHASE1
                                        STARTS.
      ERRORFILES <ERRORTABLE1>,<ERRORTABLE2>
      CHECKPOINT <INTEGER>
DEFINE <STRUCTURE>
FILE = <FILENAME>
INSERT <STMT>
END LOADING - PHASE2 STARTS.
. LOGOFF
```

Eg 1:- Importing fixed Text data with ETL Operations.

```
.Set record unformatted;
.logon 127.0.0.1/dbc,dbc;
database VINAYAKA;
drop table epparty10;
drop table evparty10;
DELETE FROM PARTY_TGT;/* IN BETWEEN BEGIN AND END
LOADING WE CAN WRITE ONLY INSERT STMT, OUTSIDE WE
CAN WRITE ANYTHING */
BEGIN LOADING PARTY_TGT
      ERRORFILES ERPARTY10, EVPARTY10
      CHECKPOINT 10;
DEFINE
      i_cog (char(3)),
      i_cust_nbr(char(3)),
      : moturincome (char(6)),
```

CRLF (Char(2)) → Carriage Return Line feed

FILE;c:\fastload\fload\CEECA_VARTEXT /forward
UNFORMAT.txt; ⇓ enter character in i/p file.

```
INSERT INTO PARTY_TGT
(
PARTYID, PARTYEMAIL, JDATE, PARTYINCOME
)
VALUES
(
:i_pid,
tesadatavinay@yahoo.co.in',
CURRENT_DATE,
:i_partyincome
);
END LOADING;
```

Save it fastload_unformat.txt.

CEECA_VARTEXT_UNFORMAT.TXT (i/p file):-

```
ibm 468400000
tcs 567200000
cts 789400000
```

To Run it FASTLOAD SCRIPTS in Cmd prompt:-

```
FASTLOAD<C:\FASTLOAD\FLOAD\FASTLOAD_UNFORMAT.TXT
>C:\DATA_LOG\FASTLOAD_LOG.TXT
```

Eg 2:- Importing Variable Text Data:-

What type of document you get in the project?
Source Map document (or) Bussiness Specification doc.

```
CREATE TABLE PARTY4(PARTYID INTEGER, PARTYNAME
VARCHAR(30), EMAIL VARCHAR(30), PARTYCODE INTEGER,
JDATE DATE, PARTYINCOME INTEGER, USERCREATED VARCHAR(30)
DEFAULT USER) UNIQUE PRIMARY INDEX(PARTYID).
```

fastload_all_features_TGT.txt:-

```
.set record vartext ","; /* mode */
 sessions 4; /* loading data parallel & faster by default
                it is 2 in a script. */
.LOGON 127.0.0.1/dbc,dbc;
 database VINAYAKA;
 drop table Party_ERR;
 drop table Party_OV;
 DELETE FROM PARTY4;
 BEGIN LOADING PARTY4
        ERRORFILES PARTYERR, PARTYOV
        CHECKPOINT 10; /* Its act like tollgate */ /* butnumScript
ERRLIMIT 15; /* It doesn't allow errors upto 15 */
DEFINE i_partyid (varchar(10)), /* Input file struct */
       i_partyname (varchar(30)),
       i_partycode (varchar(10)),
       i_partyincome( varchar(15))
FILE=C:\FASTLOAD\fload\DATA_vartext.txt;
record 2 thru 10; /* It skips 1st row colnames */
INSERT INTO Party4
^
```

```
partyid, partyname, EMAIL, PARTYCODE, TDATE, PARTYINCO
-ME)
VALUES
(
:i_partyid,
:i_partyname,
'TERADATAYINAY @YAHOO.CO.IN',
31300,
CORRENT_DATE,
:I_PARTYINCOME
);
END LOADING;
.LOGOFF
```

input file :- DATA_VARTEXT.txt

```
PARTYID, PARTYNAME, PARTYCODE, PARTYINCOME
   1, IBM, 20, 80000
   2, TCS, 30, 60000
   3, CTS, 40, 50000
   4, ACCT, 60, 70000
   5, WIPRO, 70, 80000
   T, MURALI, 30, 70000
   3, KRISHNA, 40, 9000
   4, MUKESH, 70, 80000
SEL * FROM PARTYERR
```

OV_Table Error.

ET_Table Error.

It doesn't show any other. (log file)

Data Parcel it contains the Error record. it can be seen by one tables and see the error record.

How do you run the script. If Script(fastload) fails? If you debugging that script.

FASTLOAD LOCKS:-

A) ACCESS LOCK:-
It applied in the acquisation phase.

B) WRITE LOCK:-
It applied in the application phase.

FASTLOAD ERROR HANDLING:-

Q 1) Aborted in phase1 data acquisition incomplete?
If errors are there rectify it and simple resubmit the script. So that it runs from last checkpoint or first record.

Q 2) Aborted in phase2 data acquisition Completed?
simply take BEGIN & END LOADING statements from the script and run again. So that it runs from AMP TO TABLE.
BEGIN identify Target Table corr. to Amp, END LOAD identify Amp & load.

*** Q 3) How do you load multiple files to a table by using fastload script?
(or)
there is no END LOADING statement more data to receive?

Remove the END LOADING the statement in the Script and replace the file by one by one in the Script till last file and Submit every time. So, that the data is appended at the Amp level.
For the last file specify the END LOADING Stmt in the Script and RUN. So that it runs from AMP to TABLE.

4) FastLoad Script is failed & Error tables are available and how do you restart?

There are 2 ways.

a) In case of old file to run:

1) Don't drop error tables.
2) Don't release the locks on the target table
3) Simply rectify the error in the Script or file and run it again. So, that it runs from last configuration.

b) In Case of new file to run;

1) Drop error tables.
2) Try to release the lock on the target table.
3) run the Script with the New file freshly.

5) How do you access the table if the locks are available in the realtime?

A; Syntax:

LOCKING <TableName> FOR ACCESS
SELECT * FROM <TableName>

Eg; LOCKING PARTY FOR ACCESS.
sel * from party

Genral poblem in the Lab:-

If the Scoipt fail & if you are tsying to see the data on the table and it shows "Table is being loaded" for running the same Scoipt follow the below process.

a) Comment the droop & delete Stmts in the Scoipt

b) Rectify the error in the file or Scoipt and run again.

22/1

MULTILOAD

1. It is the utility and loads max 5 tables.
2. It is basically used for loading bulk volume of data.
3. Compare to Fastload it is slower but loads into Multiple Tables with difference DML operations.
4. It Supports max 20 DML operations.
5. Here Insert, Update, delete and upsert (update else insert) operations supported.
6. It allows data into
   a) already populated tables (data available)
   b) Set or Multiset tables.
   c) NUSI or Soft Referential integrity
7. It reduces the utilization of PE, BYNET.
8. It's load on the operation of AMP.
9. There is no transient general overhead.
10. Max 15 Multiload Scripts run at a time.

~~Navigat~~

11. It loads duplicates also but it not possible in FLOAD.

Navigation :-

a) Start → run → mload.

b) Start → programs → TD client → TD Mload.

Multiload Error Tables :-

A) ET TABLE

B) UV TABLE

c) WT Table [Work Table] - It created in the initial phase and loaded the data in acquisition phase.

d) Restart log Table - It is used for better restarting Once the MLOAD Script is failed.

MULTILOAD PHASES:-

It runs in 5 phases for its operations.

1. PRELIMINARY PHASE (INITIAL PHASE)

A. It creates one ET, UV, WT for every target table.
B. It creates one RESTART LOG Table for Every import task
C. It applies initial level locks (Acquisition, Application...) on the Target Tables.
D. It takes No. of Amp+2 Sessions.

2. DML TRANSACTION PHASE:

A. It Submits DML operations Against the Teradata Database.
B. It takes a tag for every DML operation and Submits

3. ACQUISITION PHASE:

A. It takes data from file and Load into Amp Work Tables.
B. If Errors are there Loaded into ET, UV Tables
C. It applies "ACCESS LOCK" on the target tables at this phase.
D. It is fully Restartable in this phase

NOTE: There is no Acquisition phase for Delete Task.

4. Application phase:

A. It applies the corresponding operations on the corresponding Tables.

B. It uses workTables while doing the operations on Target.

C. It applies "WRITE LOCK" on the Target table.

D. If Errors are these loaded in to ET, UV Tables.

E. IT is fully Restartable in this phase.

5. TASK CLEAN UP PHASE:

A. If all the above phases are Succeeded.

1. DROPS, ET, UV, WT, RESTART LOG Tables.

2. It Releases the LOCKS on the Target Tables.

MULTILOAD STRUCTURE:-

- LOGON < >
  - LOGTABLE <RESTARTLOG TABLE>
  - BEGIN IMPORT/DELETE MLOAD TABLES <TABLE1> ---- <TABLE5>
  - LAYOUT <LAYOUTNAME>
    - FIELD --> the value to be inserted into the database
    - FILLER --> these values are filtered out before loading to TD ROWS.
  - DML LABEL <LABELNAME> --> it is tag name for every DML operation
    <DML OPERATION> FROM M FOR N THRU K
    - IMPORT INFILE <FILENAME>
      FORMAT <TEXT/VARTEXT/BINARY/RECORD>
      LAYOUT <LAYOUTNAME>
      APPLY <DML TAG>
  - END MLOAD.

LOGOFF

NOTE:-

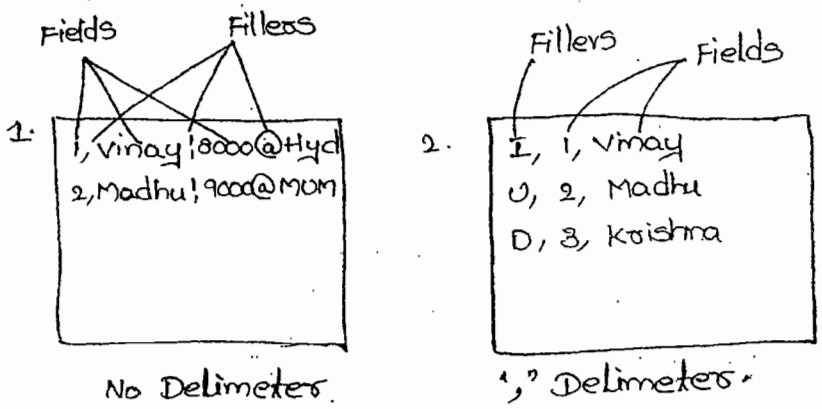
FROM M → starting position.

FOR N → The No. of Records to Read.

THRU K → Till END Record.

Eg: 6 TO 8TH RECORD.

FROM 6 FOR 2 THRU 8.



Note:-

Multiload performs 2 tasks

BEGIN IMPORT
- Insert
- Update
- Delete

DELETE → This delete works more fastly compare to BEGIN task delete.

MULTILOAD TASKS:-

There are two types of tasks in multiload.

a) IMPORT b) DELETE

a) IMPORT:-

1) Here we can perform insert, update, delete Operations
2) It supports NOSI data loading

FAQ

SET

PID (NUPI/NUSI)	PName	PLOC
1	Vinay	mum
2	Kaish	hyd
1	Madhu	hyd ✓
①	Vinay	mum ✗

Note:- Max 15 scripts fastload/Multiload run at a time.

In BTEQ there is no limit.

3) It runs in 5 phases.
4) It takes DML Tags and apply operations.

b) DELETE:-

1) It deletes only One table at a time.
2) It is also fully automatically restartable.
3) There is no transient journal overhead.
4) It runs in 4 phases.
5) It doesn't take DML tag and apply operations.

Eg1: Deleting the Data by Using Delete Task:-

this script runs in 4 phases it is DELETE TASK

```
.LOGON 127.0.0.1/dbc,dbc;
.logtable vinayaka.mno;
 database vinayaka;
.BEGIN DELETE MLOAD TABLES PARTY_TGT;
 DELETE FROM PARTY_TGT;
.END MLOAD;
.LOGOFF;
```

Save it MLOAD_DELETE.txt in C:\MLOAD\.

Run it > MLOAD<C:\MLOAD\MLOAD_DELETE.txt>C:\DATA_LOG\MLOAD_LOG.TXT.

this script runs in 5 phases it is IMPORT Task

Eg2:- Importing a data into a table by using IMPORT Task:-

This Script runs in 5 phases it is IMPORT Task

```
.LOGON 127.0.0.1/DBC,DBC;
.LOGTABLE VINAYAKA.KK1;
DATABASE VINAYAKA;
.BEGIN IMPORT MLOAD TABLES PARTY1;
    .LAYOUT VINAYLAYOUT;
        .FIELD I_PARTYID * VARCHAR(3);        (* → starting character.)
        .FIELD I_PARTYNAME *VARCHAR(30);
        .DML LABEL INS1;
         INSERT INTO PARTY1(PARTYID,PARTYNAME)
          VALUES(:I_PARTYID,:I_PARTYNAME);
        .IMPORT INFILE C:\MLOAD\VINAY_VARTEXT.TXT
          FORMAT VARTEXT ',' LAYOUT VINAYLAYOUT
          APPLY INS1;
    .END MLOAD
.LOGOFF
```

Save it MLOAD_INSERT_VARTEXT.TXT.

Input file (VINAY_VARTEXT.TXT):-

```
10,VINAYAKA
25,VINAY
39,MADHU
FT,KITTU
28,KKRKKKRRRRRRRRRRRR
```

Two Enters

Run it > MLOAD <C:\MLOAD\MLOAD_INSERT_VARTEXT.TXT>
C:\DATA_LOG\MLOAD_INSERT_LOG.TXT

FAQ Note:-

1) In Every Company minimum 200 Scripts Can be ran i.e, Scheduled jobs.

2) How do you say Script is Successfull & fail?

If Script Successfull the
Highest return code is '0'

&

If Script fails then
4 - It's low Severity
8 - It's high Severity
12 - It's high Severity

Eg3:- MLOAD_SCO1.txt (Slowly Changing Dimension)

```
.LOGTABLE VINAYAKA.Logtable001;
.LOGON 127.0.0.1/dbc,dbc;
Database vinayaka;
.BEGIN IMPORT MLOAD TABLES PARTY5;
.LAYOUT Cust_Trans;
.FIELD PID 1CHAR(1);
.FIELD PNM 2CHAR(5);
.FIELD PLOC * CHAR(3);
.DML LABEL PARTY_UPSERT
DO INSERT FOR MISSING UPDATE ROWS; /*Here insert no update*/
UPDATE PARTY5 SET PARTYNAME=:PNM, PARTYLOC=:PLOC
WHERE PARTYID=:PID;
INSERT INTO PARTY5 VALUES(:PID,:PNM,:PLOC,);
```

```
.IMPORT INFILE C:\MLOAD\DATA_FIXEDTEXT.TXT FORMAT
 TEXT
LAYOUT Cust_Trans
.APPLY PARTY_DSET;
.END MLOAD;
.LOGOFF;
```

DATA_FIXEDTEXT.TXT (I/P FILE):-

```
1,VINAYOSA → update
2,MADHUHYD → insert
```
} SCD1

1 → Enter

Eg:- Implementing SCD2:-MLOAD_SCD2.txt

```
.LOGTABLE VINAYAKA.Logtable001;
.LOGON 127.0.0.1/DBC,DBC;
DATABASE VINAYAKA;
.BEGIN IMPORT MLOAD TABLES PARTY6;
.LAYOUT Cust_Trans;
.FIELD PID * VARCHAR(30);
.FIELD PNM * VARCHAR(30);
.FIELD PLOC * VARCHAR(15);
.FIELD STD_DT * VARCHAR(15);
.DML LABEL PARTY_INSERT;
.INSERT INTO PARTY6 VALUES(:PID,:PNM,:PLOC,:STD_DT,
 '0001-01-01');
.DML LABEL PARTY_UPDATE
```

DO INSERT FOR MISSING UPDATE ROWS;...

UPDATE PARTY6 SET END_DT = :STD_DT WHERE PARTYID =

:PID;

INSERT INTO PARTY6 VALUES(:PID, :PNM, :PLOC, :STD_DT

'0001-01-01',);

.IMPORT INFILE C:\MLOAD\DATA_VARTEXT_2.txt FORMAT

VARTEXT ','

LAYOUT cust_Tools

APPLY PARTY_Update APPLY PARTY_INSERT;

.END MLOAD;

.LOGOFF

DATA_VARTEXT_2.txt (4p file);—

1, VINAY, US, 2011-01-20

2, KRISHNA KANTH, hyd, 2011-01-20.

*** WorkTable To See: SHOW TABLE WT_PARTY1

CREATE MULTISET TABLE VINAYAKA.WT_PARTY1, NO FALLBACK

NO BEFORE JOURNAL, NO AFTER JOURNAL,

(

Field1 VARBYTE(64) FORMAT 'X(64)' NOT NULL)

PRIMARY INDEX(Field1);

MULTILOAD LOCKS:-

1) ACCESS LOCK:-

Applied in the acquisition phase

2) WRITE LOCK:-

Applied in the application

3) ACQUISITION LOCK:-

Available in the initial phase till Acquisition phase during this operation any DML include Drop can be performed.

4) APPLICATION LOCK:-

It is available from initial phase to Application phase during this Operation SELECT Operation including DROP can be performed.

5) EXCLUSIVE LOCK:-

Used for releasing the Locks on Target Tables.

MULTILOAD ERRORHANDLING:-

1) While Executing multiload, client System restarted?

A: we need to manually submit the Script so that it loads data from last checkpoint.

2) For Executing multiload Script, teradata Server restarted?

A: Along with the Server multiload Script will be restarted and runs from last checkpoint.

3) If multiload Script fail in acquisition or applic-ation phase then how do you restart?

A: There are 2 ways in case of 1) old file to own.

a) rectify the Errors in the Script or file.

b) Dont Drop ET, UV, WT & Log tables and don't release Locks on Target tables Simply resubmit the script.

2) New file to own.

a) Drop the ET, UV, WT & restart LogTable.

b) Release the Logs on the Target tables

3) Specify new file in the Script own it.

4) How do you release the Locks On Target tables?

A: 1) We release the Locks On the Target tables with diff. Syntaxes & diff. phases.

2) These Syntaxes will work only in BTEQ Utility.

ACQUISITION : RELEASE MLOAD <TABLE1>, <TABLE2> -- <TABLE5>
PHASE

APPLICATION; RELEASE MLOAD <TABLE1>, <TABLE2> -- <TABLE5>
PHASE
IN APPLY

Slowly Changing Dimension:-

Type 1 (No History) :- ① New Record inserted
② Existing Record updated/Modified.

Daily Table

3	Krishna	Hyd
1	Vinay	USA

History

PN	PNAME	PLOC
1	VINAY	Hyd
2	Madhu	MDM

SCD Type

PNo	PName	Ploc
1	Vinay	USA
2	Madhu	Mum
3	Krishna	Hyd

Note:- Only 1 location for 1 customer

SCD Type 2 (Current, Expired):-

① New record inserted with status current.
② Old record ⓐ inserted with status current.
ⓑ Update previous record status expired.

Daily

1	Krishna	Hyd
2	Vinay	USA

History

PNo	PName	Ploc	Status
1	Vinay	Hyd	current
2	Madhu	Mum	current

Type 2

History

PNo	PName	Ploc	Status
1	Vinay	Hyd	Expired
2	Madhu	Mum	current
3	Krishna	Hyd	current
1	Vinay	USA	current

specifies 1 is in that place

SCD Type 2 (Effective Date Range) [Realtime using this type]:-

Daily

3	Krishna	Hyd
1	Vinay	USA

History

PNo	PName	Ploc	Stdate	Enddate
1	vinay	Hyd	22/01	NULL
2	Madhu	Mum	22/01	Null

Type 3

History

PNo	PName	Ploc	Start	Enddate
1	vinay	Hyd	22/01	23/01
2	Madhu	Mum	22/01	Null
3	Krishna	Hyd	23/01	Null
4	Vinay	USA	23/01	Null

specifies 1 is in that location

26/1

TPUMP

[TERADATA PARALLEL DATA PUMP]

1) Some people called as Trickle pump.
2) It loads data into 60 or more tables (64)
3) It supports insert, update, delete and upsert operations.
4) It is also fully restartable.
5) It loads the data even if the table is
   a) Set or Multiset
   b) SI or RI (Referential Integrity)
   c) Empty or populated
6) It applys locks at row level.
7) It does processing ~~block by block~~. packet by packet.
8) It is also checkpoint configurable.
9) REALTIME USAGE:-
   a) To load less volume of data into multiple tables.
   b) To load near real time data (latest data, generally in hours)

TPUMP LOCKING:-

1) It applies locks at row level so concurrent users can work on the same table simultaneously.
2) Script (Tpump) can be stopped and locks can be released without any ill effect.

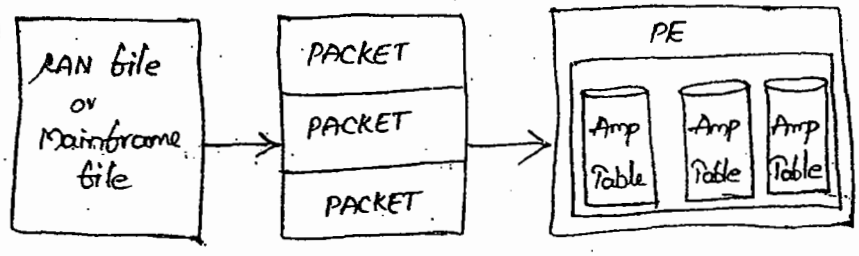
TPUMP TABLE:-

1) One Error table for Every Target table.

2) One Lock table for TPUMP Script.

TPUMP EXECUTION PROCESS & STRUCTURE:-

1) It process packet by packet of input file data



2) Tpump structure is similar to Multiload

3) Tpump executes by creating a Macro for every DML operation..

TPUMP ADDITIONAL COMMANDS:-

Session	number	Number of Sessions.
Errortable	tablename	Specify the name of the Error table
ErrLimit	Errcount	Limit on rejected records. If exceeded, Tpump performs a checkpoint and terminates.
checkpoint	frequency	Number of minutes b/w checkpoints
Tenacity	hours	Number of hours TPump will retry logon for requested sessions.
Sleep	Minutes	Number of minutes to wait after failed logon before attempting another logon.
NOTIFY	level	Level to notify exit.

Serialize	off/on	ON guarantees that operations on a given key occurs serially. Requires KEY parameter on appropriate FIELD Definition.
PACK	No. of stmts	Number of statements to pack into a multi statement request (600 max
PACKMAXIMUM		Dynamically determines the maximum PACK value.
RATE	statement rate	Number of stmts sent per minute. If unspecified, the rate is unlimited.
NOMONITOR		TPump will not check for statement rate change.
LATENCY	seconds	Number of seconds delay before a partial buffer is sent to the database
ROBOST	off/on	OFF signals TPump to use 'simple' restart logic (restart at last checkpoint).
MACRODB	dname	The database where TPump will store its macros.

Error Table Structure is similar to the Errortables in the multi-load and it contains the below columns.

a) Errorcode
b) Error field
c) Data parcel
d) Source Sequence (this the row where the error in the i/p file)

Importance keyword of PACK:-

It improves the throughput performance when it loading from network.

Increasing the pack will increase the no. of statements sending so that it loads the data faster.

Importance of ROBUST ON:-

It is a default Option of Tpamp and avoids the reapplying the rows that have already being processed in the Event of restart.

• Name Parameter:-

It is used to set the Utility variable name. $SYSTOBNAME with a job name of up to 16 characters.

Importance of keyword on SERIALIZE:-

It is meaning full only when a primary key for the loaded data is specified by using the key option of field Command.

Eg 1:- PUMP1.txt (Implementing filteration of data by using filters)

```
•LOG TABLE Vinayaka.Logtable001;
•LOGON 127.0.0.1/DBC,DBC;
DATABASE VINAYAKA;
•NAME rme_party1;
•BEGIN LOAD [ERRORTABLE et_party1
SLEEP 5 CHECKPOINT 5
SESSIONS 8 ERRLIMIT 2 PACK 2
TENACITY 4 ]→ Optional
SERIALIZE ON;

•LAYOUT Cust_Trans;
•FILLER Tcode1 VARCHAR(1);
•FIELD PID * VARCHAR(30) KEY;
•FIELD PNM * VARCHAR(30);
•FIELD PCODE * VARCHAR(5);

•DML LABEL Party_Delete;
DELETE FROM Party1 where PARTYID=:PID;

•DML LABEL Party_Update
DO INSERT FOR MISSING UPDATE ROWS;
UPDATE PARTY1 SET PARTYNAME=:PNM,
PARTYCODE=:PCODE WHERE PARTYID=:PID;

INSERT INTO PARTY1 VALUES(:PID,:PNM,:PCODE);
```

tata

```
.IMPORT INFILE C:\TPUMP\DATA_VARTEXT.TXT FORMAT
 VARTEXT ',';

LAYOUT Cust_Trans
APPLY Party_Delete where Transcode='D'
APPLY Party_Update where Transcode='U';
.END LOAD;
.LOGOFF;
```

DATA_VARTEXT.txt (I/p file)

```
D,1,KKKK,20
U,8,KRISHNA,30
U,2,VINAY KONNA,30
```

Note:-

.NAME nme_party1 ;- where developer executed DML operations for every 1 DML operation it creates 1 macro to identify the macro we give this name.

TENACITY :- It is for create connection in hours to the servers.

PACK2 ;- It Executes 2 staments then 2 statements - - -

SERIALIZE ;- Your telling to the system it is the primary index Column and it loads fastly.

Eg2:- Importing simple file data (TPUMP_IMPORT_SIMPLE.TXT)

```
.Logtable vinayaka.Cust;
.Logon 127.0.0.1/dbc,dbc;
database vinayaka;
.Name nme_party1;
.BEGIN LOAD SESSIONS 4;
.Layout P1;
.field partyid * Varchar(2);
.field partyname *varchar(30);
.DML LABEL ins;
Insert into party1(partyid, partyname) Values(:partyid,
 :partyname);
.Import infile c:\tpump\mfest_varstext.txt format
 varstext ','
 Layout P1 APPLY ins;
.END LOAD;
.LOGOFF;
```

mfest_varstext.txt (i/p file):-

```
80,madhu
40,Ramatha
```

Eg3:- Implementing SCD Type 2 Current-Expired meth

```
Create Table Partys(partyid integer, partyname varchar
(30), partyloc varchar(15), status varchar(30))
insert into partys(1,'VINAY','HYD')

.LOGtable vinayaka.logtable001;
.Logon 127.0.0.1/dbc,dbc;
 Database vinayaka;
.Name nme_party1;
.Begin Load sessions 4;
.Layout Cust_Fears;
.FIELD PID * VARCHAR(30);
.FIELD PNM * VARCHAR(30);
.FIELD PLOC * VARCHAR(5);

.DML LABELINS;
INSERT INTO PARTYS(:PID,:PNM,:PLOC,'Current');

.DML LABEL UPD;
UPDATE PARTYS SET STATUS='Expired' WHERE PARTYID=:PID;

.IMPORT INFILE C:\TPOMP\DATA_VARTEXT_SCD.TXT FORMAT
 VARTEXT ','
LAYOOT Cust_Fears
APPLY UPD
APPLY INS;
.END LOAD;
.LOGOFF;
```

DATA_VARTEXT_SCD.txt:-

```
2,MADHU,MUM)
1,VINAY,USA
```

TPUMP	MULTILOAD.
1) Applies locks at rowslevel	1) Applies locks at table Level.
2) Loads max 64 tables	2) Loads max 5 tables.
3) packet by packet processing	3) Block by Block processing
4) Creates Macros for every DML operation as part of its process.	4) Runs in 5 phases.
5) too less volume of data we need to go for it.	5) too huge amount of data generally we go for it.
6) Supports SI & RI	6) Doesn't Support USI & Hard Referential integrity

TPUMP ERRORHANDLING:-

1) There is a load to the table every 1 hour & 24/7, morning traffic is high, Afternoon traffic is less, Night traffic is high. Acc. to Situation which utility you used how do you load?

Ans TPUMP is Suggestable is here.
By using the packet size increasing or decreasing we can handle the traffic.

2) TPUMP is failed and error tables are existed then how do you restart?

Ans: There are 2 ways.
a) In Case of old file to run

1) dont drop ET, log tables and the macros.
2) dont release the locks on the target tables.
3) rectify the error and script on file and run it.

b) In case of new file to run.

1) Drop ET, Log table, Created Macros
2) Release the row level locks on the target tables
3) Run the script with the new file.

FAST EXPORT

1) It Exports large Volumes of the data from tables to files.
2) It is fully automatic restartable [BTEQ doesn't support].
3) It can take internal as well as external parameters
4) Max 15 fastexport scripts we can run at a time.
5) It uses various environmental variables such as sessions, Tenacity, Sleep etc;
6) It supports outmod (o/p module) programming in the script.
7) It process block by block data.

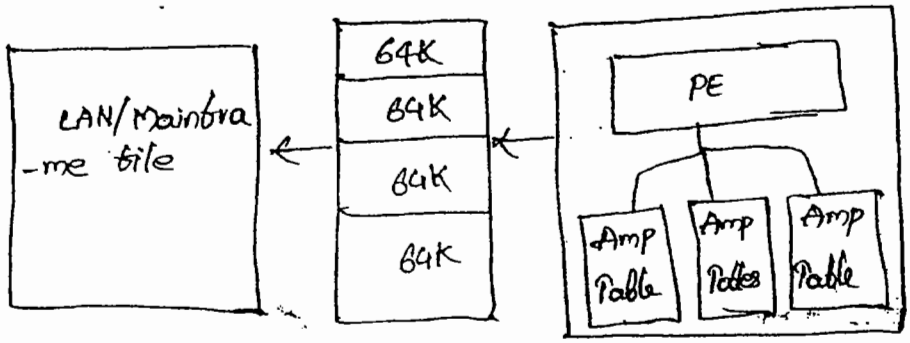
Navigation:-

a) Start → Run → Fexp

b) start → programs → TD client 13.0 → Teradata fastexport

FASTEXPORT PROCESS & EXECUTION:-

It process block by block from tables to the file



1) It applies dirty Read lock on the table and mo
2) -ves the data to Amp. (unordered)
Table → Amp (unordered)
(store records) (distribute)
3) Once the Complete data available on the amp it releases the lock on the table and loads to the file. (unordered) Amp → File (unordered)
Redistribute

Note:- If you need Sorted data Exporting then Amp has to redistribute, the datawise so the tera -data recovering so that don't take ORDER BY clause in SELECT statement.

FASTEXPORT STRUCTURE:-

```
.LOGON
.LOGTABLE <DATABASENAME.TABLENAME>
    .BEGIN EXPORT SESSIONS MIN/MAX
                  TENACITY HRS
                  SLEEP    MINS
                  NOTIFY OFF/LOW/MEDIUM/HIGH
    EXPORT OUTFILE <FILENAME>
           DD      <DDNAME>
           [MODE RECORD/INDICATOR]
           [BLOCKSIZE BYTE]
           [OUTLIMIT INTEGER]
           [FORMAT TEXT/VARTEXT/BINARY/UNFORMAT
                   /FASTLOAD - - -]
           [MLSCRIPT <SCRIPTPATH>]
```

<SELECT STATEMENT TO EXPORT DATA>

- END EXPORT
- LOGOFF

EXPORTING INTO MULTIPLE FILES:

- LOGON
  - BEGIN EXPORT <>
    - EXPORT OUTFILE <FILENAME1>
      <SELECT STMT1>
    - END EXPORT
  - BEGIN EXPORT<>
    - EXPORT OUTFILE <FILENAME2>
      <SELECT STMT2>
  - END EXPORT
- LOGOFF

PASSING PARAMETERS:-

There are two ways

1. Internal parameters: By using .SET command with in the script we use the parameters.
2. External parameters: From external file we get the parameter data.
   1). ACCEPT → It takes only one record from file.
   Eg: 'VINAY'
   2). IMPORT → It takes one by one record from the file.
   Eg: 'vinay'
   'madhu'
   'krishna'

LIMIT :- It limits the no. of o/p records.

MLSCRIPT :- This is the multiload script created by the fastexport

FORMAT :- format of the file i.e, exporting.

BLOCKSIZE :- Defines the maximum blocksize to be used in returning exported data. default is 64K.

MODE :- It takes either record mode or Indicator Mode

FORMAT OPTIONS :- The below format options supported in fast export.

a) FASTLOAD - a two-byte integer, n, followed by n, followed by n bytes of data, followed by an end-of-record marker, either x '0A' or x '0D'

b) BINARY :- a two-byte integer, n, followed by n bytes of data.

c) TEXT - an arbitrary number of bytes followed by an end-of-record marker, either x '0A' or x '0D'

d) UNFORMAT - exported as it is received from CLIV2 without any client modifications.

CLIV2 - CAL INTERFACE VERSION 2

Eg 1:- Exporting data from at table [FASTEXPORT_INITIAL.TXT]

```
•LOGON 127.0.0.1/dbc,dbc;
•logtable vinayaka.test_log1;
•begin export tenacity 4 sleep 2;
•export outfile c:\data_log\fastexport_party1_data.out   <-- o/p file (binary data)
mode record format text mlscript c:\data_log\mlfast
.export.txt; /* Create mload script */
select * from vinayaka.party;
•end export;
•logoff;
```

Eg 2:- Exporting data in variable text format [',' delimeter]

[FASTEXPORT_VARTEXT.txt]

```
•RON FILE C:\LOGON.TXT;
•logtable vinayaka.test_log1;
•begin export tenacity 4 sleep 2;
•export outfile c:\data_log\fastexport_party1_data.out
 mode record format text;
 Select
 CAST(PARTYID AS VARCHAR(10))|| ';'||
 CAST(PARTYNAME AS VARCHAR(30))|| ';'||
 CAST(PARTYLOC AS VARCHAR(30)) AS OUTPUT_DATA
 from vinayak.party;
.end export;
•logoff;
```

T.:-

127.0.0.1/DBC,DBC;

e it in a 'c' drive. In Realtime we don't need on everytime by default it create in one drive and Run it in our script.

Run it as

fexp <C:\fexp\fastexport_vartext.txt> C:\DATA_LOG\fastexport_vartext_log.txt.

Eg3:- Exporting the data into multiple files and using internal parameters.

Create multiset table party, Fallback (Organization varchar(30), partyid Integer, partyname varchar(30), partyloc varchar(30), partyincome Integer, pdate varchar(30))

Insert into party('Capgemini', 1, 'vinay', 'Hyd', 80000, '2011-01-19');

Insert into party('IBM', 2, 'Madhu', 'pune', 70000, '2011-01-20');

Create multiset Table Test(Testid Integer, Test_desc varchar(50), partyid Integer, partyname varchar(30))

insert into Test(1, 'IT', 1, 'vinay')

insert into Test(2, 'IT', 2, 'Madhu')

```
.RUN FILE c:\LOGON.TXT;
.logtable VINAYAKA.Log1;
.Set partyname1 to 'VINAY';
.display 'Run1 started at --&systime' to file c:\
 DATA_LOG\fastexpoot_track.txt;
.begin expoot tenacity 4 sleep 2;
.expoot outfile c:\DATA_LOG\ PARTY_DETAILS.TXT mode
 record format text;
 select
 CAST (PARTYID AS VARCHAR(10)) ||',' ||
 CAST(PARTYNAME || '_' || PARTYLOC as varchar(30)) ||',' ||
 CAST(PARTYINCOME AS VARCHAR(30))
 ||',' ||
 Cast(CASE
        WHEN PARTYCODE=10 THEN 'IT'
        WHEN PARTYCODE=20 THEN 'HR'
        ELSE 'SALES AND MARKETTING'
 END as varchar(30))
 ||',' ||
 Cast((CAST(JDATE AS DATE)(FORMAT 'DD-MM-YYYY')) as
 varchar(10)) ||',' || cast(TRIM(ORGANIZATION) as varchar
 (30)) from VINAYAKA.party;
 .end expoot;
 .display 'Run2 started at -&systime' to file
  c:\DATA_LOG\fastexpoot_track.txt;
 .begin expoot tenacity 8 sleep 2;
 .expoot outfile c:\DATA_LOG\TEST_DETAILS.TXT mode
  record format text;
```

```
Select * from VINAYAKA.test where partyname =
'$partyname1';
.end export;
.LOGOFF.
```

Eg4:- Exporting the data into a file by Taking Values from external file:- [fastexport_accept]

```
.RUN FILE C:\LOGON.TXT
.logtable VINAYAKA.log2;
.ACCEPT partyname1, partyincome1 from file c:\fexp\
 daily_data.txt;
.begin export tenacity 4 sleep 2;
.export outfile c:\DATA_LOG\PARTY_DETAILS.TXT mode
 record format text;
 select
CAST(PARTY1D AS CHAR(10)) ||
CAST(PARTYNAME AS CHAR(20)) ||
CAST(PARTYINCOME AS CHAR(10)) ||
CAST(PARTYCODE AS CHAR(10)) ||
CAST(JDATE AS CHAR(10)) AS PARTY_DETAILS
from VINAYAKA.party where partyname='$partyname1'
OR PARTYINCOME > $ partyincome1;
.end export;
.Logoff
DAILY_DATA.txt;

'RAJESH'  20000
```

Limitations of fastexport :-

a) SELECT stmt should not contain non-data table reference.

Eg :- Current_Date → non-data table.

b) you cannot take equality condition for primary index or USI.

c) with option to do subtotal and full-total.

d) 'USING' modifier to submit data parameters as a constraint to select.

OLELOAD

[Object Linking and Embedding LOAD]

It is used for to move the data from any data source, files to any datasource, files.

Eg:-

Database ↔ Database.

Database ↔ File.

File ↔ File.

DB's → Oracle, Sql Server, TD.

Files:→ flatfile, xml file, Excel files.

Eg:- Moving data from Oracle database Employee table to vinayaka teradata database as employee table.

Start → pgms → TD client 13.0 → oleload.

Select the driver (Source)

Microsoft OLE DB Provider for Oracle (MSDAORA)

Specify Serevername - Oracle

Username - Scott

password - tiger

[OK]

Select the destination

Teradata Ole load.

Teradatahost → localtd.

userid → DBC

pswd → DBC

~~Create settings~~ > click ok

> select Emp tables

> Come to Settings → Table Name showing "emp" - ok

> click launch → select Utility.

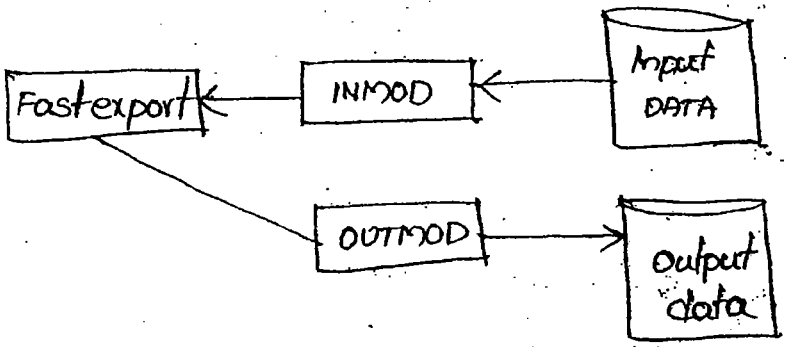
Teradata Warehouse builder:-

1. It is used for moving data from database, file to another database, file.

INMOD & OUTMOD Routines:-

INMOD for Input module programming

OUTMOD for Output module programming



INMOD: Read input data values from a file.

2) It quality's select request usually we use with multiload, Tpump etc; import facilities only

OUTMOD:

1) It process Answerset data

2) It modify's, discards the record responses usually more applicable like Fastexport

PERFORMANCE TUNNING

To identify the bottlenecks and resolving it we go for performance tunning.

bottleneck : It is not an error but causes the system delaying in performance.

Eg:-

There is a query or script which is suppose to run in 2 minutes but it ran for halfanhour and finally it is succesceded. In that situation we need to identify the bottleneck and resolve it.

Identifying BottleNecks:-

We use EXPLAIN REQUEST MODIFIER OR TERADATA VISUAL EXPLAIN OR PERFORMANCE MONITOR TOOLS

EXPLAIN REQUEST MODIFIER (EXECUTION PLAN COMMAND):-

It displays the below information

this info. is provided by the optimizer.

a) No. of cpus it is cooking
b) Amount of time it is taking
c) No. of Amps it is using
d) the type of join strategies it is using
e) The amount of spool memory its occupying

Syntax:-

EXPLAIN <SQL QUERY>

Eg:-

a) Explain Sel * from party;

b) Explain Delete from party;

NOTE:-

If the statstics are not collected then optimizer displays either "low confidence data" or "NO confidence data" by using random sampling information.

COLLECT STASTICS:-

Stastics nothing but 'DEMOGRAPHICS OF THE TABLE' storing in it seperate place.

Eg:-

No. of rows, max. row size, memory left, No. of cols etc for the table.

If the stastics are collected then the optimizer gives the least expensive plan (less cost plan) that the query is going to executed by the database.

It prevents random sampling information.

If the table is updated we need to update the statistics then only we get least expensive plan with high confidence.

Statistics are collected on Columns or Index.

Eg:-

help stats party /* To see already collected stattics.

collect statistics on party column(partyid);

Syntax:-

collect statistics on <tablename> index/column(colname)

before collect statistics

explain select * from party;

o/p: low confidentiality to be 4 rows.

after ~~statistics~~ collecting statistics.

Collect statistics on party index(partyid);

explain SELECT * FROM PARTY;

o/p: high confidentiality to be 8 rows.

Note: To see the stats collected.

Syntax; HELP STATS <tablename>.

Note:- Refer the doc for all confidencies.

Pseudo Table Lock:-

It prevents deadlock when multiple users are trying to work on the same table and same Amps.

Cost Based Optimization:-

Here the cost in the sense time (Some times the. of rows can also be considered)

Less cost plan will process in shortest path, Huge cost plan will process in longer path.

Confidence Levels:-

A) At Single table level

1) No Confidence:

→ If there are no statics on the column or index sets specified

→ There are complex expressions and aggregations are there but no statistics are executed

2) Low Confidence;

→ There are the conditions on the query on an index set that has no statistics

→ There are the conditions in the query on an index set or column set which is ended with ANDed (or) ORed with other non statistics columns.

→ The confidence for the single amp statistical operations are low.

3) High Confidence;

→ These are statistics on the indexes or columns and there is no skew (duplicate rows can go into same amp (or) uneven distribution)

B). Confidence Level for Join Operations:-

1) No Confidence:-

→ one (or both) relation in the join does not have statistics on the join columns.

2) Low Confidence:-

→ On the joining columns if one is having low confidence automatically entire is low confidence (There should not be no confidence)

3) High Confidence:-

→ If one column is having high confidence it the other one is high or join index automatically it encounter.

A) Index Join

→ There is a unique index on the join columns.

→ There is foreign key relationship b/w join columns.

Using Appropiate indexes:-

A) Use PI for better storage, access, join operations.

B) Use PPI to get good performance in terms of range based data.

c) Use SI to prevent full table scan and getting fast retrieval of data.

d) Use join, hash indexes when you're working with multiple tables for better performance.

JOIN INDEX:-

** full table scan:- Every row can touch the Every Amp.

Usage:-

1. If a join query frequently used
2. If table is used in the joining operation with other tables we need join index.

SINGLE TABLE JOIN INDEX:-

1) It is used an alternative access path to the base table.

2) In this case some or all columns of the one table with the primary index generally being foreign key in that

table then we go for single table join index.

Syntax:-

Create join index <index name> as <join query>

Eg:-

a. Singl table

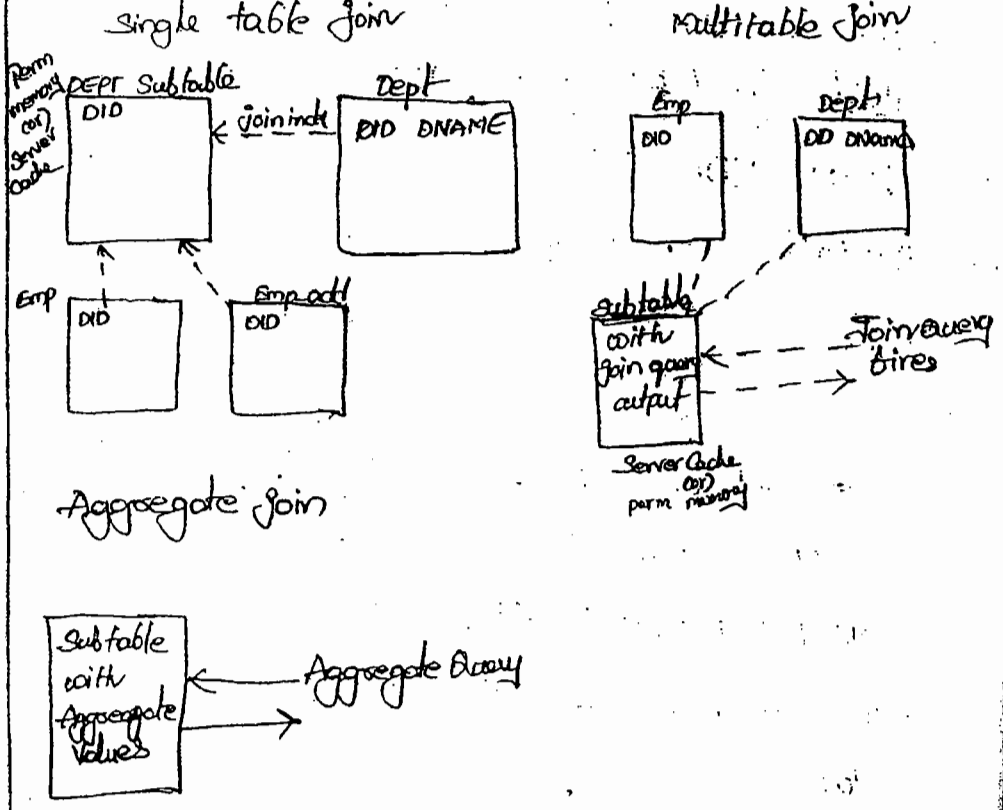
Create join index idx as sel * from dept;

multitable join index:-

1) It pre joins multiple tables and stores the result

Eg:-

Create join index jdx as select d.deptid, d.dname, e.eid from dept d inner join emp e on d.did = e.did;



Aggregate join

DTS [DATA TRANSFORMATION SYSTEM] :-

For moving data from one DB to another Interadata, as well as one Source System to Another Source System they are using this process.

In this case

→ BTEQ (INteradata)

→ Oleload/SSIS [For one Source System to other Source System]

Note:-

Dummy data by created by as at time of developet or client will give the dummy data upto 10 rows they Copy and give for testing purpose in any database.

AMS:- Here Customers, Agents, policies etc; info. maintain.

PAYT:- Here commissions, & payup commissions, payables & percentages etc; info. these.

GOLDMINE:- Here info regarding ULip's, high premiums Customer information.

WEGA :- Group insurance related info. stored.

Creating a view for Datamart:-

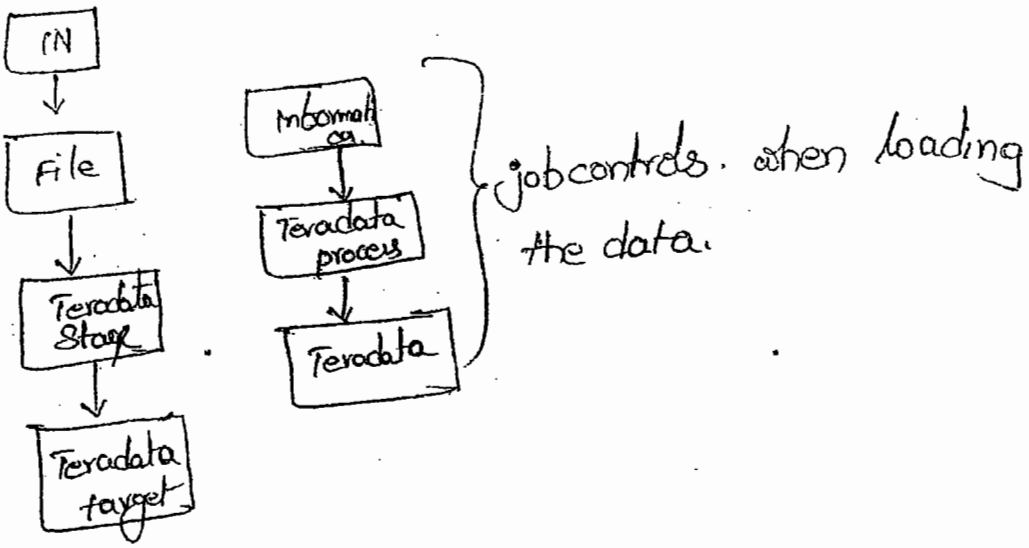
Viewname - Riskmgmt Datamart.

used - 62 tables

Duration - 6 months

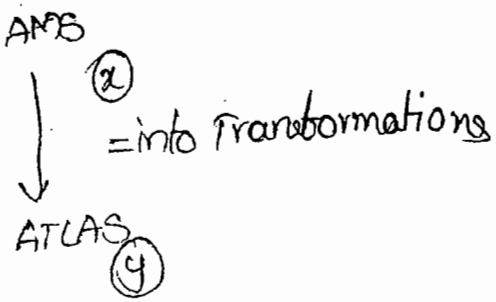
input - many business doc's

Job Control



Direct loading

Transformation loading



Q:- ATLAS LIFE ASIA Direct loading.

AGENT_ID ⟶ AGNTNUM

CLIENT_ID ⟶ CLNTNUM

* Any company the teradata model are.

1) Normalized

2) Dimensional Model

- star
- snowflake.

** You'll never get a chance to create table until you are Admin until unless you create Stage table, Scara table etc.

Dimension Tables Names:

Dim Agreement-Status.

Dim CalendeDate, DimDBAName.

** First you load data in Dimension table then load into Fact tatable.

Fact Tables Names:

FactMoMngPrdctPrfl.

Column Names in Target.	Source Segment Code
CommercialPackage_Id	CommercialPackage.
InsuredCompany_Id	Company
InsuredName_Id	UnstructuredName
Calendar_Dt	ETL

insert into FactMoMgtPkod AS Select cp.cp_id, c.c_id

From CO---cp, C----c where cp.x=c.x

the load the values into Fact table in there only

two columns common.

Take only one part either extract, Transform, loading & Explian in interview.

for each there is maximum 1½ year experience.

Resume

Name;
Email id;

Professional Summary;-

Total 3+ - - - - relevent 2+

Achievements;

Certifications;

Academic Qualifications

→ BSc From O.U with 87%
→ SSC with 90, 10+2 with 85

Technical skills;-

Only relevant & additional
Technical skills add,
Don't add unnecessary skills
Ex;- OS; win/unix

Project handled / Recent Accomplishments;

project 1;-

Name;
client;

Duration;

Tools used;

Teamsize;

Role played;

Description; - - -
- - - 5/6 lines.
- - -

Responsibilities/Contribution;-

---- 6 to 7 min.
---- with respect to project.

- - - - etc;

Additional details;

PAN; ____

passport; ____

DOB; ____

Languages known; ____

Declaration;- ____
____
____

BI- Business Intelligence

Anand

ROLES & RESPONSIBILITIES:-

→ According to the doc's created & manipulated BTEQ, FASTLOAD, MULTILOAD Scripts.
→ Having knowledge in TPUMP, FASTEXPORT.
→ Implementing in performance Tuning at Various levels.
→ Use appropiate indexes such as PI, SI, JI, HI etc;
→ Worked with errorshandling by using ET, UV, WT Tables.
→ Worked extensively with SQL queries Sub queries, joins, set operations and functions.
→ Did data reconciliation across source systems.
→ Answered Adhoc request of the client by using functions, Analytical functions etc;
→ created Unit test cases and implemented Unit test results.
→ Having good knowledge in DWH & BI concepts.

① What are your day to day activities?

A: I check my mails for task allocation. Some Tasks are day-wise and some are weekwise.

② Generally the tasks are getting business doc's, understanding, Analyzing, if bugs are there intimating later once the bugs are rectify then working with same business doc's.

Getting Scripts or Queries for Testing.

Sometimes doing unit testing for existing Scripts or changed Scripts.

Sometimes working with excel tables.

Sometimes monitoring the jobs running and reporting.

Sometimes eliminating the duplicate records, multiple open records etc;

What type of documents are we getting in the realtime.

Generally we are getting Source map doc (or) target to source doc's. These are in excel sheets. Generally Contains info

a) Source structure including field def's & sizes

b) target

c) Source to target mapping rules or business rules etc;

How do you do unit testing in a project & why do we need?

As a developer i need to test the code whatever i dev -eloped. In this case i am writting some testcases in excel sheet and attaching results also. for ensuring my code is developed correctly

TestcaseNum TestCaseDesc ExpectedResult ActualResult Pass/fail

— — — — —

NOTE:- Refers the above two doc's in the material.

④ How do you perform data reconsilation (Verification)

In this process we are checking no. of rows the source & target. & need validate or check it wheather loaded successfull or not.

⑤ What is your teradata Warehouse Model.

Ans:-

A:- Currently i am working with banking/insurance project and my warehouse model is 3NF. Earlier i work with star schema, Snowflake schema models also.

⑥ What is the Warehouse approach ~~follow in your pro~~ and Lifecycle model in your project?

Inmon approach & waterfall model (Spiral, iterative, 3a3 model.

⑦ Lifecycle Steps:-

a) Req's Analysis

b) Design

c) Coding
d) Testing
e) implementation
f) maintenance
g) SLA support (Service level agreement)

) What is the testing model using in your project?

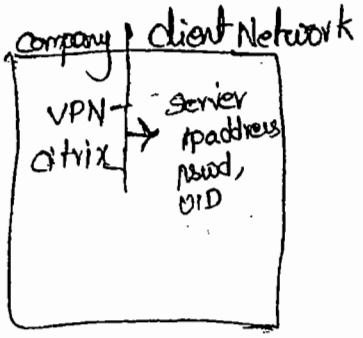
;- V model. Here testing against development will performed.

⑤ What are your testing stages having in your project?

A;- ① Unit testing (individual Unit of code tested)
② Developers performed (White box testing this testing)
③ System Testing (or) System Integration Testing (it is functionality Testing this can also be called as Blackbox Testing) → Quality Analyst will do this. [Testers]
④ User Acceptence Testing (or) End users Testing : clients perform this.

⑨ How do you connect your Servers?

A;- There are many ways i connected
a) By using VPN or Citrix Server by Specifying userid, password, and ip address of the Server.



Every Comany having two networks.

Eg:- Connect to Yahoo Server, Gmail Server

b) By using Remote desktop (RDC) [It needs OSID, PSWD, IPaddress] In this case the desktop which is created in the Remote opened as the local desktop.

(10) How do you get the files to load it & what are the file sizes?

A:- We are not getting files through mails because generally they are MB's to GB's. Client copied the files in the shared location or common location of unix and provides path to us.

(11) What are the general doc's deliver to client or Team Lead?

A:- The task whatever you allocated either the Scripts whatever i wrote or the queries or Some times of the Unit test cases.

(12) What is the Granularity of your project?

A:- Some tables daywise, Some tables weekwise or month-wise etc;

3) How many Nodes & Amps having in your project?

A:- 4 (or) 8 Nodes, 12 (or) 16 Amps.

4) What are datawarehouse size.

A:- 6 Terabytes. (big project).

other Answer:- 2 to 3 (Small)

5) What are the versions you work with Teradata?

A:- Right now i work with Teradata 12.

Earlier i worke with V_2R_5 & V_2R_6.

6) Tell Some Table Names, Mapping rules (or) Transformation rules in your project?

A:- Refer to the document you provided.

7) What are the versioning tools in your project?

A:- VSS [Visual Source Safe]

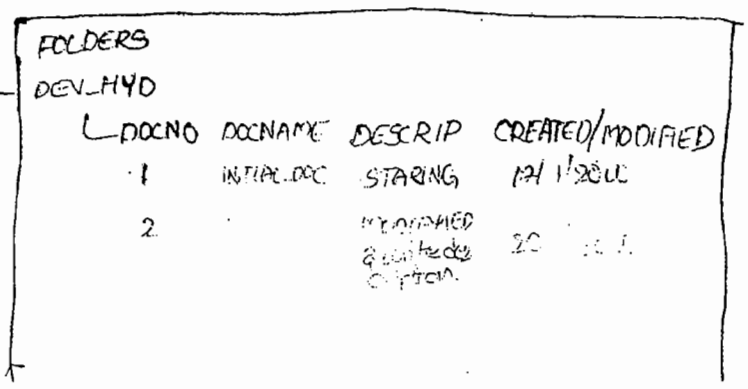
PVCS

CLEARCASE - - - - - -

Here we do doc's in check in & check out.

Open VSS

click



(18) What are the Scheduling tools you are having or using in your project?

A:- As a developer iam not responsibiles for scheduling in the project. Support people will take care, i think they are using TRILLIUM

Other Scheduling Tools:-

CONTROL M, AUTOSIS,

UNIX schedulers like AT, CRONTAB.

(19) How many Scripts you developed in the total project?

A:- for 3 years based around 50 to 60 Scripts to developed & manipulating

(20) What are the challenges you faced in your project?

(or)

Can you tell me typical issues in

A:- Generally at the time of history maintence, multiple open records, identifying duplicates and resolving it i face problems.

(21) For unknown Answers in interview?

A:- a) I didn't get a chance to work it but if you give me chance to me I ~~explorate~~ pop up.

(or)

b) No idea but basically we are looking with HASTCOA etc; explian (Sdu)

DWH & BI Questions;-

1. What is a DataWarehouse.
2. DWH approaches?
3. What is a DataMart & Types of Datamarts?
4. Diff: b/w Dependen & Independent?
5. What is the dimensions & Types
6. What is the Fact table & Exp. Types
7. What is the Measure & Exp Types.
8. What is the Factless table
9. Diff b/w star & Snowflake Schema.
10. Explian Slowly changing dimensions Available?
    TYPE1, TYPE2, TYPE3
11. What is a SURROGATE OR DWH KEY?
    It provides Uniqueness for the Rows & used to Replace the use of primary key

Genral Database Question:-

1. Diff b/w Sub query & Corelated Sub Query.
2. Diff b/w Union & Unionall
3. find 2nd max. Salary.
4. Display top 5 Salaries.
5. How do you identifies duplicate & How to resolve it
6. Diff b/w Rownumber & Rank.
7. Diff b/w Rowid & Rownumber.
8. Explian about Normalization & Normal forms.

General Unix Commands:

1. WC(L,C,--)
2. LS
3. GREP
4. UNIQ
5. SORT
6. FGREP, EGREP
7. PS
8. SED
9. TEE
10. CAT
11. CP, MV, RM

12) COT

13. TRANSCATE.

1) Display the number of lines where

ADVANCED OLAP FONCTIONS:-

RANK AND QUALIFY:-

RANK: RANK() OVER(PARTITION BY COLNAME ORDER BY COLNAME ASC/DSC)

EG1:

SEL PARTYID, PARTYCODE, PARTYINCOME, RANK() OVER(ORDER BY PARTYINCOME ASC) FROM PARTY;

EG2;

BASED ON PARTYCODE .I REQUIRE RANK

SEL PARTYID, PARTYINCOME, PARTYCODE, RANK() OVER(PARTITION BY PARTYCODE ORDER BY PARTYINCOME ASC) FROM PARTY;

EG3; 1

FIND OUT TOP 5 EMPLOYEE SALARIES

SEL PARTYINCOME, RANK() OVER(ORDER BY PARTYINCOME DESC) AS R FROM PARTY QUALIFY R<=5 GROUP BY PARTYINCOME;

ROW_NUMBER;-

IT DISPLAYS UNIQUE VALUE FOR EACH ROW;

EG: TO DISPLAY UNIQUE RANK VALUES FOR THE EMPLOYEES

SEL PARTYID, PARTYINCOME, PARTYCODE, ROW_NUMBER() OVER (ORDER BY PARTYINCOME ASC) FROM PARTY;

EXTRACT COMMAND:

IT EXTRACTS DAY, MONTH, YEAR ETC.. FROM DATE COLUMN.

SYNTAX:

EXTRACT(DAY/MONTH OR YEAR FROM DATE)

EG: SEL JDATE EXTRACT(MONTH FROM JDATE) FROM PARTY;

FORMAT COMMAND;

IT IS USED TO CHANGE THE FORMAT OF INTEGERS, STRINGS AND DATES

EG: SEL PARTYINCOME, PARTYINCOME(FORMAT '$99,999.99')

FROM PARTY;

SEL JDATE, JDATE(FORMAT 'YYYYBMMMM/DD') FROM PARTY;

TOP:

DISPLAYS TOP 5 EMPLOYEE SALARIES.

SEL TOP 5 PARTYINCOME FROM PARTYORDER BY PARTYIN-COME DESC;

Note:- csom also used for generating sequence.

(At display time it can generate sequence)

FORMATTING INTEGERS:-

9 → digit

Z - 1 Zero Suppressed digit

$

#

,

/ etc;

FORMATTING STRINGS:-

X → Single character

B → Blank space.

FORMATTING DATES:-

YYYY → full years.

YY → Year

MMMM → Full Month Name

MM → Month.

DD → Day

B → Blank space.

/
,
} Seperators.

Eg:-

To display top 5 salaries in 5 ways they are.

ORDER BY, TOP, ROW_NUMBER, Simple query

Q: Difference between rowid and rownumber?

Rowid :-

It is a physical value which is of 64 bits for identifying every row uniquely in the Amp.

Row_number :-

It is a logical displayable value to display unique for every row while showing the output.