# Teradata

AN INTRODUCTION

# What is Teradata?

- Best Data warehouse & analytical database
- Fully scalable relational database management system
- It has a capacity of Teradata bytes in size which makes it a market leader in data warehousing application
- It is one of the largest systems in the world
- It was an extension for mainframe systems

# Teradata components

1

## Processor Chip

- **BRAIN** of the Teradata system

2

## Memory

- **HAND** of Teradata system

3

## Hard Drives

- **SPINE** of Teradata system

# Why Teradata?

- Large capacity
- Performance
- Single version of truth
- Manageable growth
- Fault Tolerance

# Advantages of Teradata

- Scalability
- Parallel Efficiency
- Execution of complex queries with a maximum of 256 joins
- Support Ad-hoc queries
- Support concurrent users

# Teradata Competitive Advantages

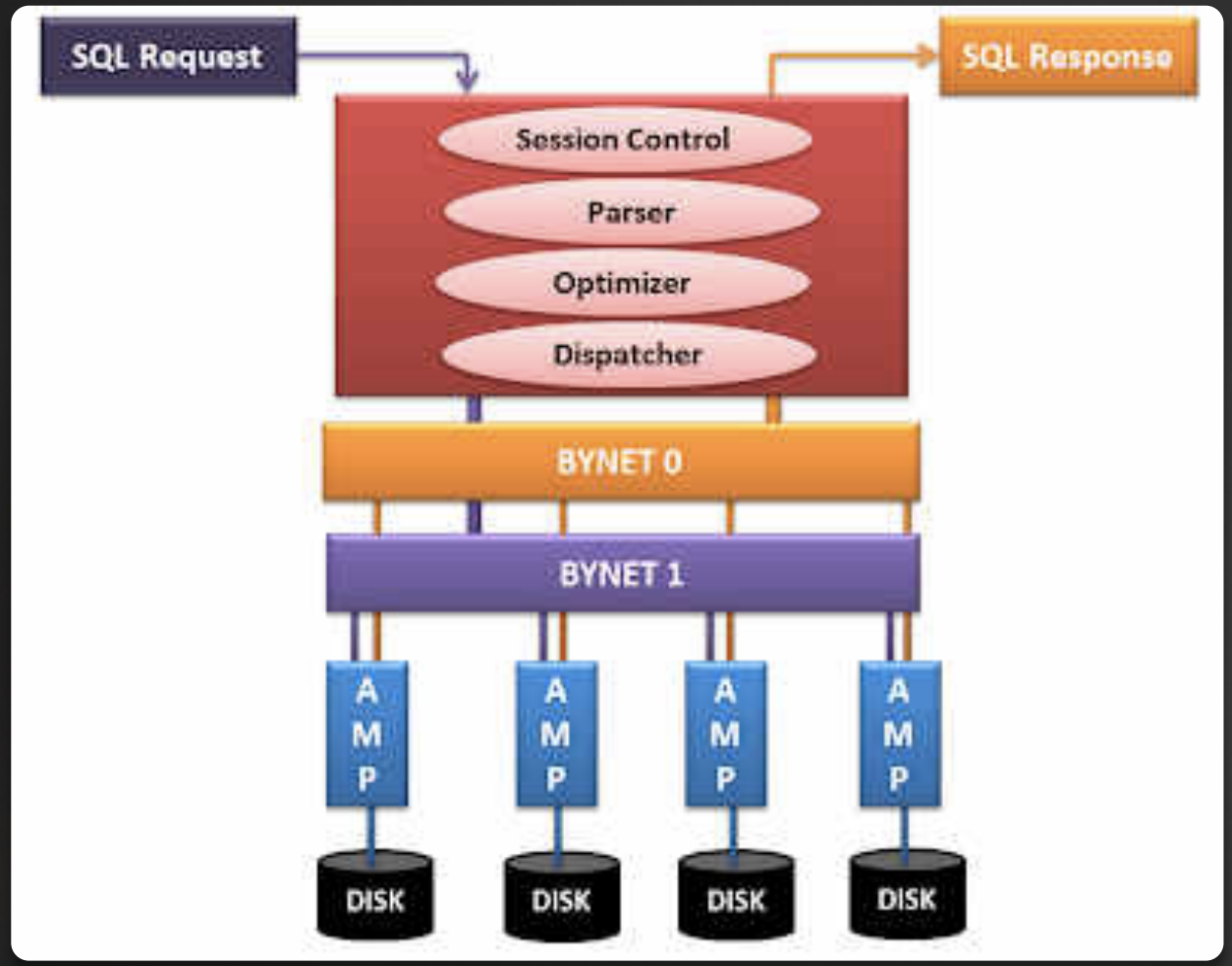
- Linear Scalability
- Unlimited Parallelism
- Mature Optimizer
- Easy Setup
- Load & Unload utilities

# How Teradata tracks object?

- *Teradata Assigns each Object a Unique Numeric ID such as* database names, table names and column names
- Each Table, View, Macro, Trigger, Stored Procedure, User-defined Function, Join Index, and Hash Index is assigned their own globally unique numeric ID.
- Each Column is assigned a numeric ID that is unique within its Table ID.
- Each Index is assigned a numeric ID unique within its Table ID.
- The Data Dictionary keeps track of all the SQL names and their associated numeric IDs.
- The PE uses the Data Dictionary to verify names and then the PE convert them to IDs.
- The AMPs use the numeric IDs supplied by the Parsing Engine in its Data Dictionary RESOLVER logic. The PE Resolver resolves that the object names exist in the Data Dictionary.

# Indexes

- Unique Primary Index
- Non Unique Primary Index
- Multi Column Primary Index
- No Primary Index Tables




# Parsing Engine

# Session Control

- The **Session Control** component verifies the request for session authorization (user names and passwords), and either allows or disallows the request.

# Parser

- The **Parser** does the following:
  Interprets the SQL statement received from the application.
- Verifies SQL requests for the proper syntax and evaluates them semantically.
- Consults the Data Dictionary to ensure that all objects exist and that the user has authority to access them.

# Optimizer

- Cost-based and develops the least expensive plan (in terms of time) to return the requested response set
- Knows about system configuration, available units of parallelism (AMPs and PE's), and data demographics.
- Robust and intelligent
- Handles multiple complex, ad hoc queries efficiently.
- Parallel-aware and cost-based
- Uses full look-ahead capability

# Dispatcher

- The **Dispatcher** controls the sequence in which the steps are executed
- Passes the steps received from the optimizer onto the BYNET for execution by the AMPs

# Access Module Processor (AMP)

AMPs, called as Virtual Processors (vprocs) are the one that actually stores and retrieves the data.

- **Point to point** – message is routed to Single AMP
- **Multi-cast** – message is routed to multiple AMP
- **Broadcast** – Message is routed to all AMPs

# BYNET

- All communication between PEs and AMPs is done via the BYNET
- BYNET receives the execution plan from the Parsing Engine and dispatches to the appropriate AMP
- Each AMP or PE can use one BYNET to retrieve communication and simultaneously accept messages using the other BYNET

# BYNET Features

1. Scalable
2. High performance
3. Fault tolerant
4. Load balanced

# What is FALLBACK?

- FALLBACK is a protection mechanism used by Teradata to handle AMP failures.
- For each data row, another copy of the row is stored in a different AMP within a cluster.
- If any AMP fails, then the corresponding rows will be accessed using FALLBACK AMP.
- FALLBACK can be mentioned while table creation using CREATE TABLE statement or after the table is created using ALTER TABLE statement.

# Other Features

# Data Blocks

- Each Table Header and Data Block have the Same TableID
- Each Data Block starts with a 72 Byte Data Block Header
- AMPs Moves Their Data Blocks into Memory to Read/Write
- Data Blocks Grow until they Reach Maximum Block Size
- Blocks Continue to Split as Tables Grow Larger

# Data Protection

- Teradata uses **Transient Journal** to protect data from transaction failures.
- **Fallback** protects the table data by storing the second copy of rows of a table
- The **Down AMP recovery journal** is activated when the AMP fails and the table is fallback protected
- **Clique** is a mechanism used by Teradata to protect data from Node failures
- **Hot Standby Node** is a node that does not participate in the production environment
- Redundant Array of Independent Disks (**RAID**) is a mechanism used to protect data from Disk Failures

# Performance Tuning

- EXPLAIN
- COLLECT STATISTICS
- Proper data types are used
- Avoid explicit data conversions
- Remove unnecessary ORDER BY clauses unless required
- Spool space error is generated if the query exceeds per AMP spool space limit for that user
- Drop the temporary tables (staging tables) and volatiles if they are no longer needed.

# FastLoad

- Target table should be empty
- Only one table can be loaded using a single script
- Supports only CREATE/INSERT statement
- Doesn’t support tables with RI, SI and Triggers

# FastLoad

Phase 1

- it brings the data in 64K blocks and sends them to the target AMPs. Each AMP will then hash redistribute the rows to their target AMPs.

Phase 2

- rows are sorted by their row hash order and written into the target table.

# MultiLoad

- Target table need not be empty
- Can load/update up to 5 tables
- Supports up to 20 DML statements in single script
- Supports tables with NUSI

# MultiLoad Phases

| Phase 1 | Phase 2 | Phase 3 | Phase 4 | Phase 5 |
| --- | --- | --- | --- | --- |
| • **Preliminary Phase** – Performs basic setup activities. | • **DML Transaction Phase** – Verifies the syntax of DML statements and brings them to Teradata system. | • **Acquisition Phase** – Brings the input data into work tables and locks the table. | • **Application Phase** – Applies all DML operations. | • **Cleanup Phase** – Releases the table lock. |

# Fast Export

- FastExport utility is used to export data from Teradata tables into flat files.
- It can also generate the data in report format. Data can be extracted from one or more tables using Join.
- Since FastExport exports the data in 64K blocks, it is useful for extracting large volume of data.

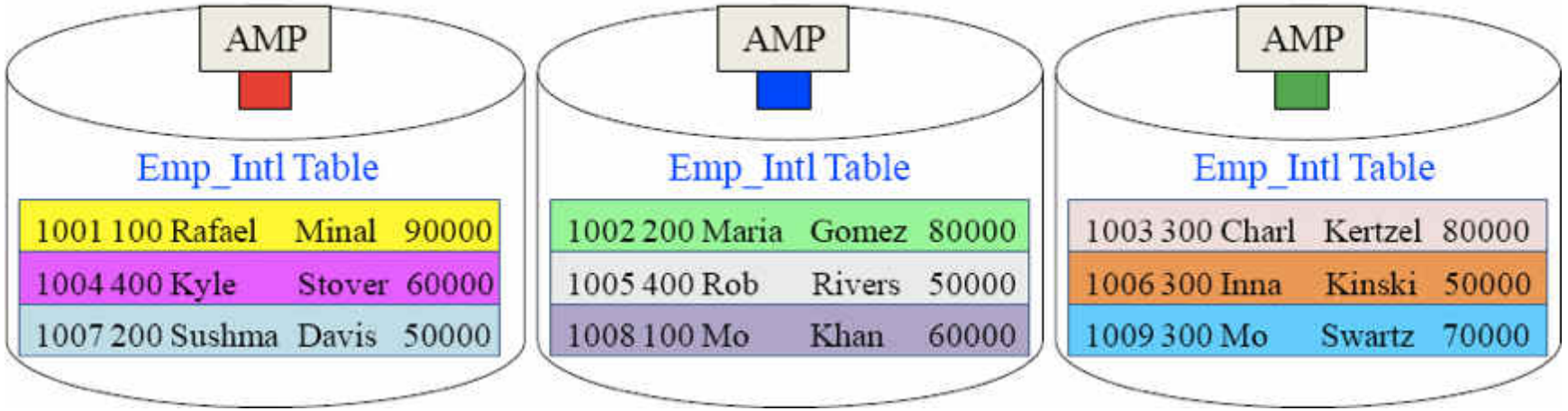
**Script: fexp < employee.fx**

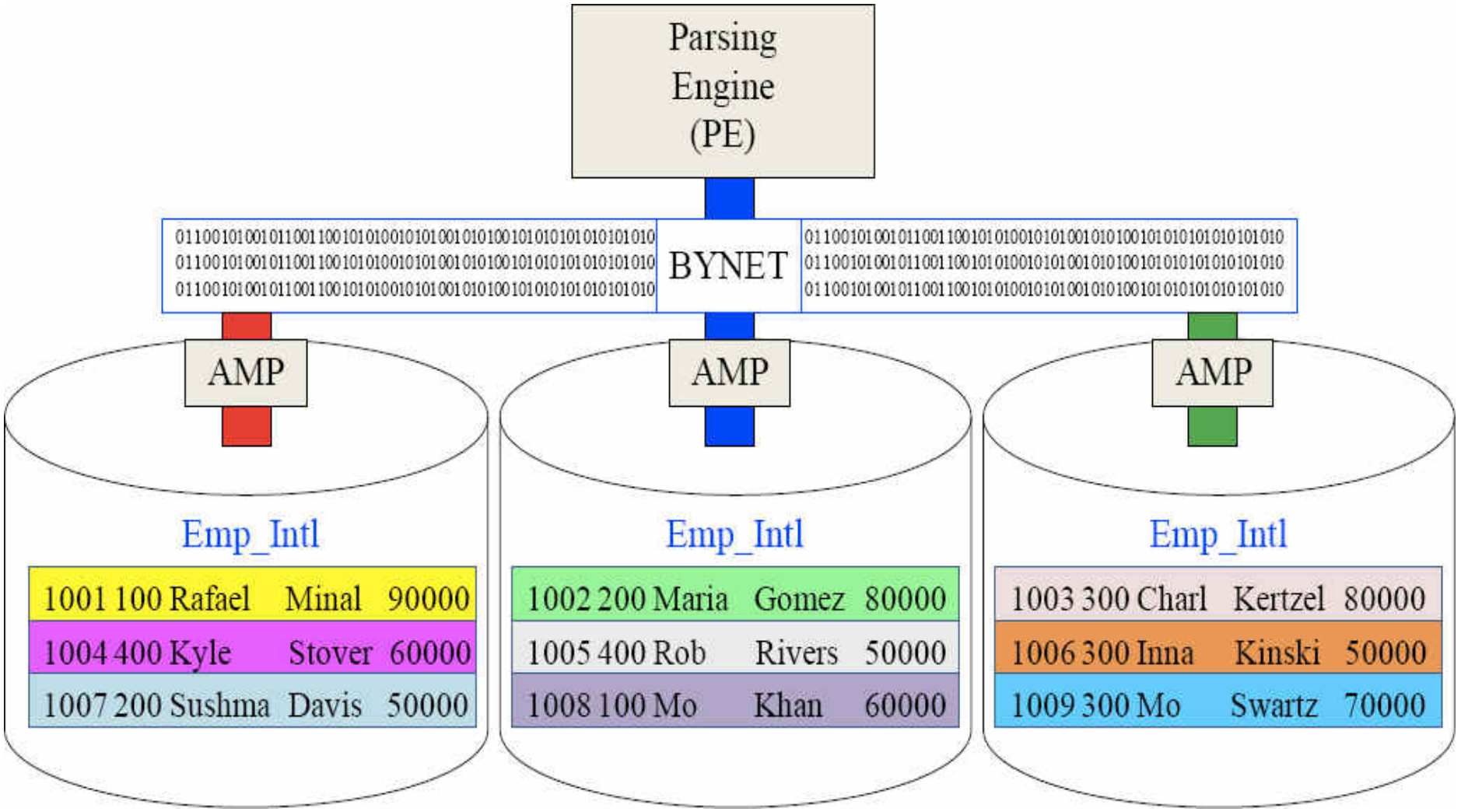
# Parallel Processing

- Pioneer in parallel processing since 1988
- Parallelism is built into its Data Structure
- Other levels of parallelism are imposed by the optimizer when building query plans
- Optimizer can choose to execute multiple steps within the same query at same time

Emp_Intl

| Emp_No | Dept_No | First_Name | Last_Name | Salary |
|---|---|---|---|---|
| 1001 | 100 | Rafael | Minal | 90000.00 |
| 1002 | 200 | Maria | Gomez | 80000.00 |
| 1003 | 300 | Charl | Kertzel | 70000.00 |
| 1004 | 400 | Kyle | Stover | 60000.00 |
| 1005 | 400 | Rob | Rivers | 50000.00 |
| 1006 | 300 | Inna | Kinski | 50000.00 |
| 1007 | 200 | Sushma | Davis | 50000.00 |
| 1008 | 100 | Mo | Khan | 60000.00 |
| 1009 | 300 | Mo | Swartz | 70000.00 |




Parsing
Engine
(PE)
BYNET
AMP
AMP
AMP
Emp_Intl
1001 100 Rafael Minal 90000
1004 400 Kyle Stover 60000
1007 200 Sushma Davis 50000
Emp_Intl
1002 200 Maria Gomez 80000
1005 400 Rob Rivers 50000
1008 100 Mo Khan 60000
Emp_Intl
1003 300 Charl Kertzel 80000
1006 300 Inna Kinski 50000
1009 300 Mo Swartz 70000

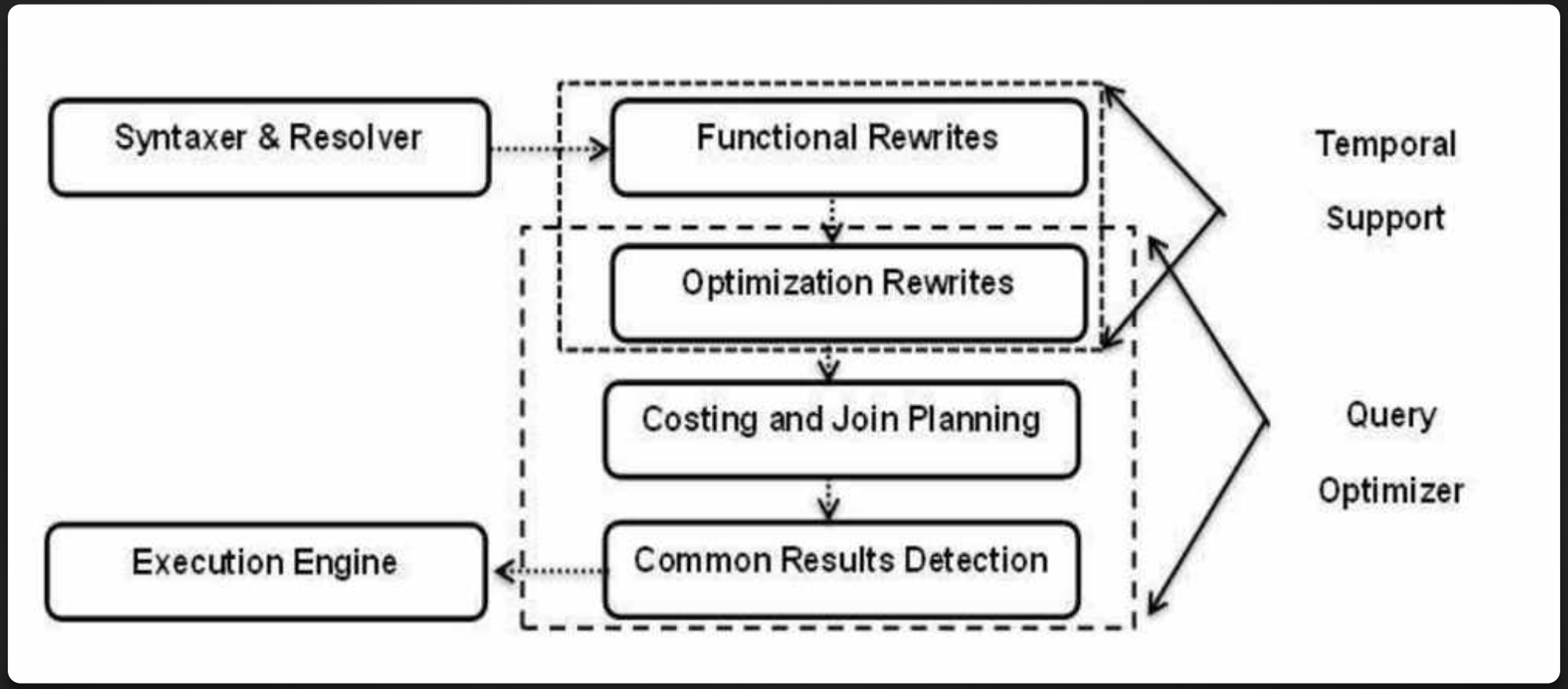
# Linear Scalability

- There is no limit to Linear Scalability
- Add AMPs to double its speed for Linear Scalability
- It is also known as '**DIVIDE and CONQUER**' theory, according to which we are dividing the work equally and getting the result faster.

# Temporal Locking

- Temporal feature is based on the TSQL2 specification
- Two common approaches to support temporal query processing in a database engine i.e. functional query and native support.

# Temporal support in Teradata



# Temporal constraints

- Can be enforced at the level of each table and between tables.
- Temporal constrains across tables represent temporal referential integrity
- Temporal referential integrity can be flexibly defined with the child table being either temporal or regular table.
- def

# Other facts

- Primary keys are not mandatory in Teradata whereas Primary Index is mandatory.
- Data distribution is based on Primary Index value.
- Primary keys doesn’t accept NULLs whereas Primary Index accepts NULL values.
- Primary keys are unique whereas Primary Index can be either unique (UPI) or non unique(NUPI).
- Primary keys doesn’t change whereas Primary Indexes change.

| Data Types | Length (Bytes) | Range of values |
|---|---|---|
| BYTEINT | 1 | -128 to +127 |
| SMALLINT | 2 | -32768 to +32767 |
| INTEGER | 4 | -2,147,483,648 to +2147,483,647 |
| BIGINT | 8 | -9,233,372,036,854,775,80 8 to +9,233,372,036,854,775,8 07 |
| DECIMAL | 1-16 | |
| NUMERIC | 1-16 | |
| FLOAT | 8 | IEEE format |
| CHAR | Fixed Format | 1-64,000 |
| VARCHAR | Variable | 1-64,000 |
| DATE | 4 | YYYYYMMDD |
| TIME | 6 or 8 | HHMMSS.nnnnnn **or** HHMMSS.nnnnnn+HHMM |
| TIMESTAMP | 10 or 12 | YYMMDDHHMMSS.nnnnnn **or** YYMMDDHHMMSS.nnnnnn +HHMM |

# Data Types in Teradata

# Thank You!

## Questions?