Base table

RID	item	city
R1	H	V
R2	C	V
R3	P	V
R4	S	V
R5	H	T
R6	C	T
R7	P	T
R8	S	T

Item bitmap index table

RID	H	C	P	S
R1	1	0	0	0
R2	0	1	0	0
R3	0	0	1	0
R4	0	0	0	1
R5	1	0	0	0
R6	0	1	0	0
R7	0	0	1	0
R8	0	0	0	1

City bitmap index table

RID	V	T
R1	1	0
R2	1	0
R3	1	0
R4	1	0
R5	0	1
R6	0	1
R7	0	1
R8	0	1

Note: H for "home entertainment," C for "computer," P for "phone," S for "security," V for "Vancouver," T for "Toronto."

Figure 3.15 Indexing OLAP data using bitmap indices.

that value. In contrast, join indexing registers the joinable rows of two relations from a relational database. For example, if two relations $R(RID, A)$ and $S(B, SID)$ join on the attributes A and B, then the join index record contains the pair (RID, SID), where RID and SID are record identifiers from the R and S relations, respectively. Hence, the join index records can identify joinable tuples without performing costly join operations. Join indexing is especially useful for maintaining the relationship between a foreign key[3] and its matching primary keys, from the joinable relation.

The star schema model of data warehouses makes join indexing attractive for cross-table search, because the linkage between a fact table and its corresponding dimension tables comprises the foreign key of the fact table and the primary key of the dimension table. Join indexing maintains relationships between attribute values of a dimension (e.g., within a dimension table) and the corresponding rows in the fact table. Join indices may span multiple dimensions to form **composite join indices**. We can use join indices to identify subcubes that are of interest.

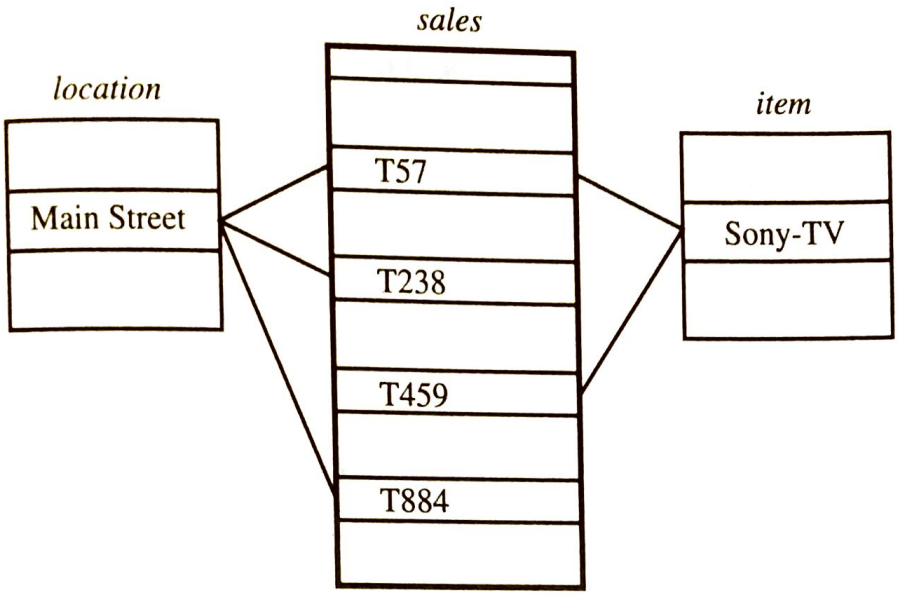


Figure 3.16 Linkages between a *sales* fact table and dimension tables for *location* and *item*.

Join index table for *location/sales*

location	*sales_key*
...	...
Main Street	T57
Main Street	T238
Main Street	T884
...	...

Join index table for *item/sales*

item	*sales_key*
...	...
Sony-TV	T57
Sony-TV	T459
...	...

Join index table linking two dimensions *location/item/sales*

location	*item*	*sales_key*
...	...	...
Main Street	Sony-TV	T57
...	...	...

Figure 3.17 Join index tables based on the linkages between the *sales* fact table and dimension tables for *location* and *item* shown in Figure 3.16.

Suppose that there are 360 time values, 100 items, 50 branches, 30 locations, and 10 million sales tuples in the *sales_star* data cube. If the *sales* fact table has recorded sales for only 30 items, the remaining 70 items will obviously not participate in joins. If join indices are not used, additional I/Os have to be performed to bring the joining portions of the fact table and dimension tables together. ■

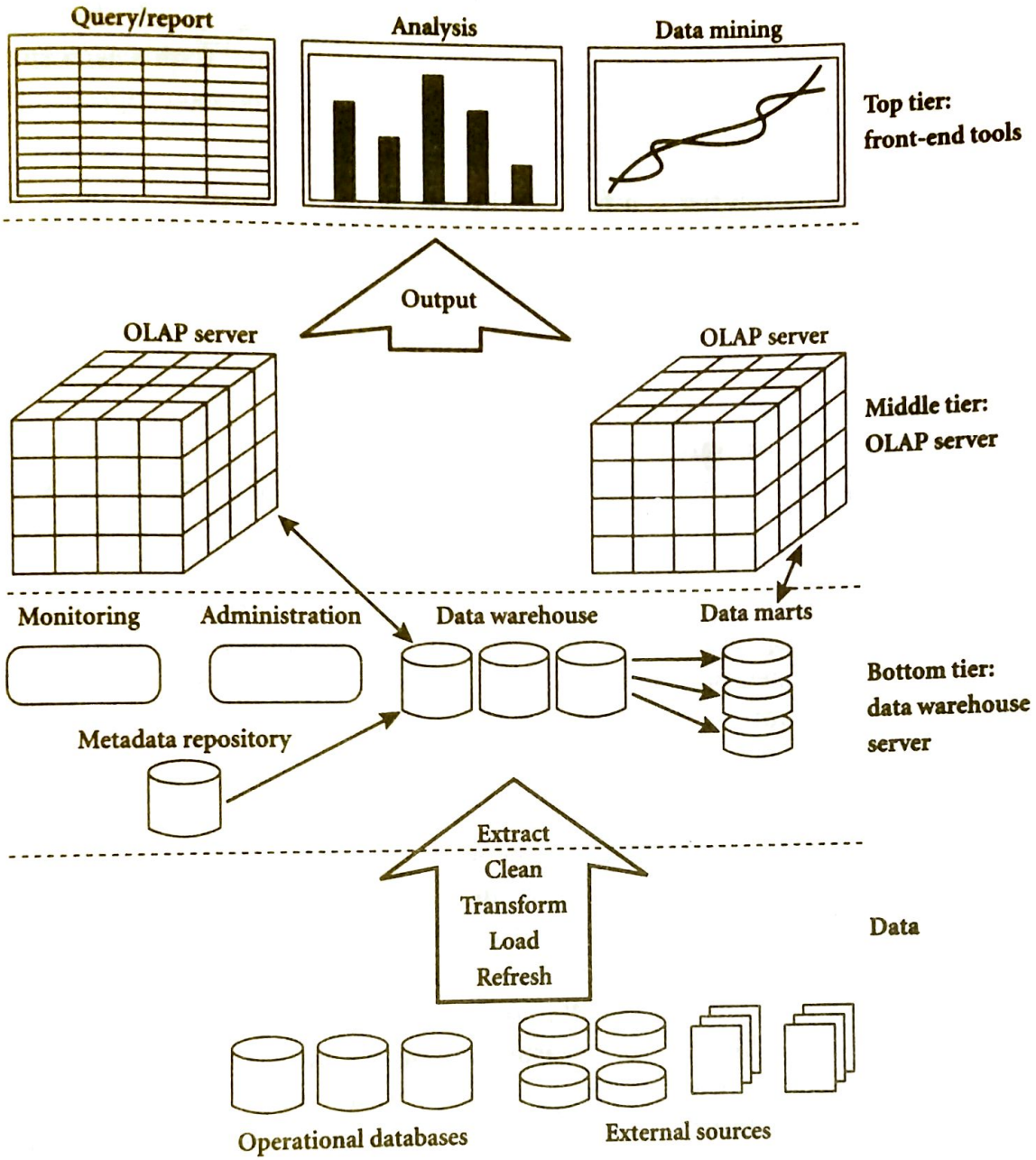


Figure 3.12 A three-tier data warehousing architecture.

and units-sold.

A three-tier Data Warehouse Architecture:

(1) The bottom tier is a warehouse database server that is a relational database system. Back-end tools and utilities are used to feed data into the bottom tier from operational databases or other external sources. These tools and utilities perform data extraction, cleaning and transformation, as well as load and refresh functions to update the data warehouse

The data are extracted using application program interfaces known as gateways. A gateway is supported by the underlying DBMS and allows client programs to generate SQL code to be executed at a server. This tier also contains a metadata repository, which stores information about the data warehouse and its contents.

2. The middle tier is an OLAP server that is typically implemented using either a (1) relational OLAP model (i.e an extended relational DBMS that maps operations on multidimensional data to standard relational operations.

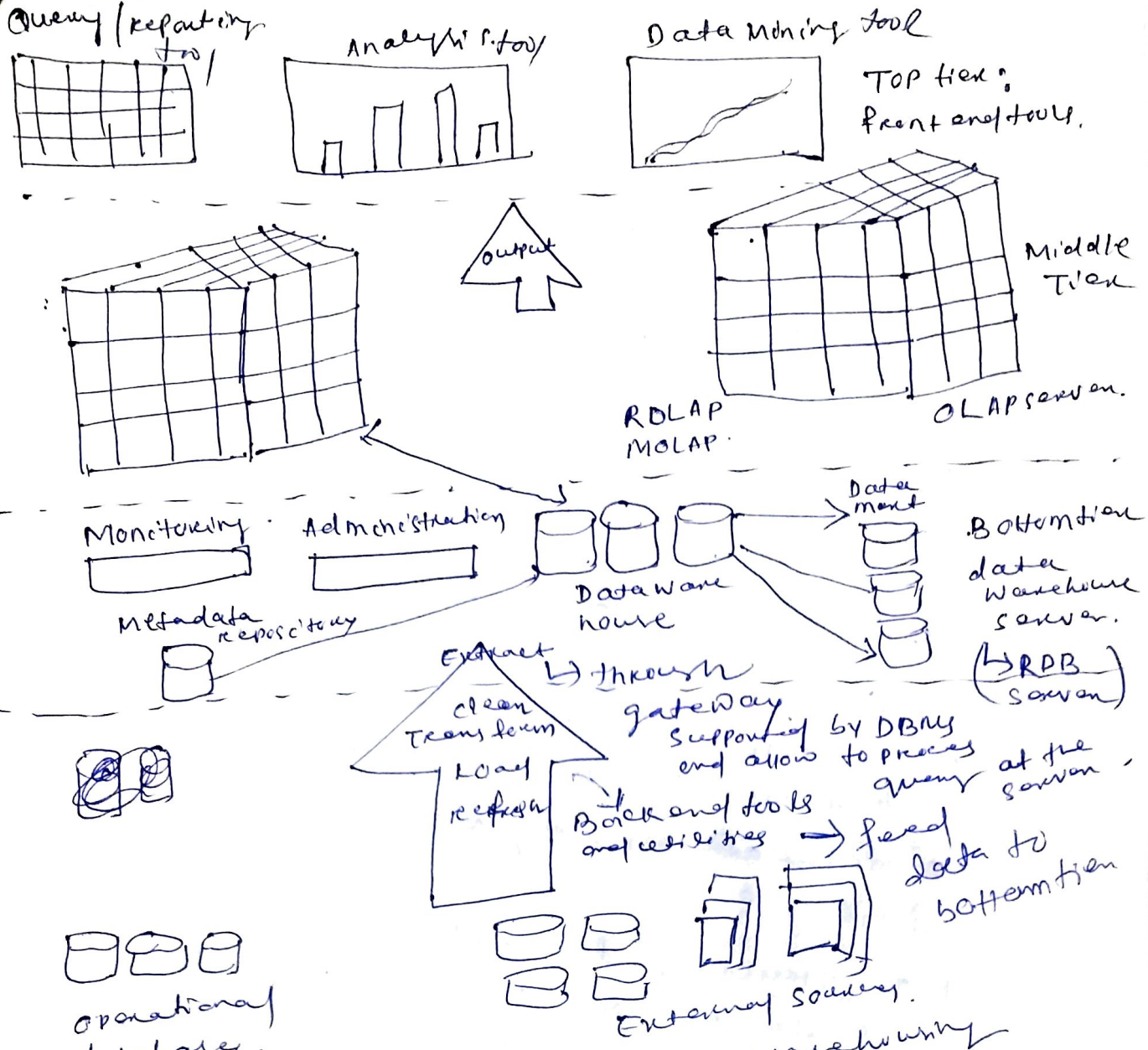
or (2) a multidimensional OLAP (MOLAP) model, i.e, a special purpose server that directly implements multidimensional data and operations.

(3) The top tier is a front end client layer, which contains query and reporting tools, analysis tools, and/or data mining tools.

From the architecture point of view, there are three data warehouse models:

(1) Enterprise warehouse
(2) Data mart.
(3) Virtual warehouse

Enterprise warehouse → collects all of the information about entire organization. It contains detailed data as well as summarized data.
Its size is from a few gigabytes to hundreds of gigabytes, terabytes etc. An enterprise data warehouse may be implemented on traditional mainframe, computer super servers etc.
It requires years to design and build.



A three tier data warehousing architecture.

Data Mart

A data mart contains a subset of corporate wide data that is of value to a specific group of users.

Ex: A marketing data mart may confine its subjects to customer, item and sales.

The data contained in data marts are summarized data.

Data marts are implemented on low cost departmental servers.

The implementation requires weeks rather than months or years.

it is of two type.

Independent data marts → sourced from data captured from one or more operational systems or external information providers or from data generated locally within a particular dept. or geographic area.

Dependent data marts → sourced directly from enterprise data warehouses.

Virtual warehouses → it is a set of views over operational databases.

Metadata Repository

Metadata are data about data. When used in a data warehouse, metadata are the data that define warehouse objects.

A metadata repository should contain the following :

(1) The structure of the data warehouse : which includes the warehouse schema, view, dimensions, as well as data mart locations and contents.

(2) operational metadata → which include data lineage (history of migrated data and the sequence of transformations applied to it), currency of data (active, archived or purged) and monitoring information (warehouse statistics, error reports and audit trails.

(3) The algorithms used for summarization. → which include measure and dimension definition algorithms, granularity, subject areas, aggregation, summarization and predefined queries and reports.

(4) The mapping from operational environment to the data warehouse —
which includes source database, gateway description, data partitions, data extraction, cleaning, transformation, data refresh and security.

(5) Data related to system performance
→ which include indices and profiles that improve data access and retrieval performance, rules for the timing and scheduling of refresh and replication cycles.

(6) Business ~~data~~ metadata → include business terms and definitions, data ownership and charging policies.

Data warehouse Back end tools and utilities
→ It is used to populate and refresh the data.

Data extraction → gathers data from multiple heterogeneous and external sources.

Data cleaning → which detects errors in the data and rectifies.

Data transformation → converts data from host format to warehouse format.

Load → which sorts, summarizes, consolidates, checks integrity and builds indexes and partition.

Refresh → it propagates the updates from data sources to warehouse.