Data Warehouse:

- A data warehouse is a database optimized to analyze relational data coming from transactional systems and line of business applications.
- The data structure, and schema are defined in advance to optimize for fast SQL queries, where the results are typically used for operational reporting and analysis.
- Data is cleaned, enriched, and transformed so it can act as the "single source of truth" that users can trust.

'A **complete repository** of historical corporate data extracted from **transaction systems** that is available for **ad-hoc access** by **knowledge workers**.'

Complete repository:

- *All the data is present from all the branches/outlets of the business.*
- *Even the archived data may be brought online.*

Transaction System

- *Management Information System (MIS)*
- *Could be typed sheets (NOT transaction system)*

Ad-Hoc access

- *Does not have a certain predefined database access pattern.*
- *Queries not known in advance.*
- *Difficult to write SQL in advance.*

Knowledge workers

- *Typically NOT IT literate (Executives, Analysts, Managers).*
- *NOT clerical workers.*
- *Decision makers.*

Why data warehouse?

- Data recording and storage is growing. The amount of data average business collects and stores is doubling every year.
- History is excellent predictor of the future. You use historical data, because it gives you an insight into how the environment is changing.
- Gives total view of the organization.
- Businesses demand Intelligence (BI).

Characteristics of a DWH:

- **Subject-oriented:** A data warehouse is always subject oriented as it provides information on a topic rather than the ongoing operations of organizations. Such issues may be inventory, promotion, storage, etc. Never does a data warehouse concentrate on the current processes. Instead, it emphasizes on modeling and analyzing decision-making data.
- **Integrated:** A data warehouse combines data from various sources. These may include a cloud, relational databases, flat files, structured and semi-structured data, metadata, and master data. The sources are combined in a manner that is consistent, relatable, and ideally certifiable, providing a business with confidence in the data's quality.
- **Time-Variant:** In this, data is maintained via different intervals of time such as weekly, monthly, or annually etc. The data collected in a data warehouse is acknowledged over a given period and provides historical information.
- **Non-Volatile:** As the name defines the data resided in data warehouse is permanent. It also means that data is not erased or deleted when new data is inserted. In this, data is read-only and refreshed at particular intervals.

How is data warehouse different?

- The data warehouse is different because, again it's not a database you do data entry. You are actually collecting data from the operational systems and loading into the DWH. So the transactional processing systems like the ERP system are the source systems for the data warehouse. You feed the data into the data warehouse. And the data warehouse typically collects data over a period of time.
- So if you look at your transactional processing OLTP systems, normally such systems don't keep very much history. Normally if a customer leaves or expired, the OLTP systems typically purge the data associated with that customer and all the transactions off the database after some amount of time. So normally once a year most business will purge the database of all the old customers and old transactions. In the data warehouse we save the historical data. Because you don't need historical data to do business today, but you do need the historical data to understand patterns of business usage to do business tomorrow, such why a customer left?

What are the benefits of data warehouses?

- Improves Business Intelligence
- Increases data security
- Improves data standardization and quality
- Saves time

Key difference between Database and Data Warehouse:

- A database is a collection of related data that represents some elements of the real world, whereas a Data warehouse is an information system that stores historical and commutative data from single or multiple sources.
- A database is designed to record data, whereas a Data warehouse is designed to analyze data.
- A database is an application-oriented collection of data, whereas Data Warehouse is a subject-oriented collection of data.
- Database uses Online Transactional Processing (OLTP), whereas Data warehouse uses Online Analytical Processing (OLAP). OLTP systems don't keep history. From these systems you cannot get balance statement of more than a year old. Whereas, DWH keeps historical data, even of bygone customers.
- Database tables and joins are complicated because they are normalized, whereas Data Warehouse tables and joins are easy because they are denormalized.
- ER modeling techniques are used for designing Databases, whereas data modeling techniques are used for designing Data Warehouses.

How much history?

Depends on:

- Industry.
- Cost of storing historical data.
- Economic value of historical data

Rate of update?

Depends on:

- volume of data,
- nature of business,
- cost of keeping historical data,
- benefit of keeping historical data

Stages of DWH:

Types of DWH

1. **Financial:** Financial data warehouses are often the first data warehouse that an organization builds. This is appealing because:
   Financial data is ALWAYS at the nerve center of the organization. In most organizations (but not all), financial data represents one of the smallest volumes of data that exist. Finance touches all aspects of an organization and has a common denominator i.e. money. Financial data is directly influenced by the day -to-day activities of financial processing and the list goes on.
2. **Telecommunications:** The telecommunications data warehouse is dominated by the sheer volume of data generated at the call level subject area.
   There are many ways that call level detail can be accommodated:
   - Only a few months of call level detail
   - Storing lots of call level detail scattered over different storage media
   - Summarizing or aggregating call level detail
   - Storing only selective call level detail, and so forth

Applications of DWH

1. **Fraud detection:** By observing patterns of data usage, and behavior of customer fraud can be detected very easily.
   - People have typical purchase patterns.
   - Deviation from patterns.
   - Certain cities notorious for fraud.
   - Certain items bought by stolen cards.
   - Similar behavior for stolen phone cards.

Denormalization

Denormalization is a strategy used on a previously-normalized database to increase performance. It is a database optimization technique in which we add redundant data to one or more tables. This can help us avoid costly joins in a relational database.

Why denormalization?

- **Improving query performance:** Some of the queries may use multiple tables to access data that we frequently need. Think of a situation where we'd need to join 10 tables to return the client's name and the products that were sold to them. In that case, maybe it would be wise to add a client_id attribute directly to the products_sold table.
- **Maintaining history:** Data can change during time, and we need to store values that were valid when a record was created. We could solve this problem by adding a table containing the history of these changes.

	Normalization	Denormalization
Functionality	Removes redundant information and improves data change speeds.	Combines multiple information into one unit and improves data retrieval speeds.
Focus	Cleaning up the database to remove redundancies.	Redundancies introduced for faster query execution.
Memory	Optimized and improved general performance.	Memory inefficiency due to redundancies.
Integrity	Removal of database anomalies improves database integrity.	No maintained data integrity. Database anomalies are present.
Use Case	Databases where insert, update and delete changes happen often and joins are not expensive.	Databases that are often queried, such as data warehouses.
Processing Type	Online Transaction Processing - OLTP	Online Analytical Processing - OLAP

Denormalization techniques

1. Table Splitting

Table splitting implies dividing normalized tables into two or more relations. Dividing tables happens in two dimensions:

- **Horizontally**. Tables split into row subsets.
- **Vertically**. Tables split into column subsets.

The method's **goal** is to split tables into smaller units for faster and more convenient data handling.

2. Pre-joining Tables

Pre-joined tables store the frequently used pieces of information together into one table. The objective behind pre-joining is to identify frequent joins and append the tables together in the physical data model. The process comes in handy when:

- Queries frequently execute on the tables together.
- The join operation is costly.

The method creates massive redundancies, so it is essential to use a minimal number of columns and update the information periodically.

3. Adding Redundant Columns:

This technique can be used when a column from one table is frequently accessed in conjunction with a column from another table. To avoid this join, the column is added (redundant) or moved into the detail table(s) to avoid the join.

OLAP

OLAP is Analytical Processing instead of Transaction Processing. It is also NOT a physical database design or implementation technique, but a framework. Furthermore OLAP implementations are either highly de-normalized or completely de-normalized, while OLTP implementations are fully normalized.

A data warehouse implementation without an OLAP tool is nearly unthinkable. Data warehousing and on-line analytical processing (OLAP) are essential elements of decision support, and go hand-in-hand.

MOLAP is a form of OLAP that processes and stores data directly into a multi-dimensional database.

- Adv:
    - Very high performance achieved by O(1) time lookup into “cube” data structure to retrieve pre-aggregated results.
    - Can perform complex calculations.
    - Instant response (pre-calculated aggregates).
    - Impossible to ask question without an answer.
- Disadv:
    - only limited data can be handled.
    - Long load time (pre-calculating the cube may take days!).
    - Very sparse cube (wastage of space)

ROLAP is a form of OLAP that performs dynamic multi-dimensional analysis of data stored in a relational database rather than in a multi-dimensional database.

- Adv: Greater amount of data can be processed.
- Disadv: requires more processing time/disk space.

HOLAP: HOLAP provides a combination of relational database access and "cube" data structures within a single framework. The goal is to get the best of both MOLAP and ROLAP: scalability and high performance.

- Adv: HOLAP can ‘drill through’ from the cube into underlying relational data.

DOLAP typically is the simplified version of MOLAP or ROLAP. DOLAP is inexpensive, it is fast and easy to setup on small data sets comprising of thousands of rows instead of millions of rows.

ETL

Data Transformation:

Tasks of data transformation:

- **Selection:** This takes place at the beginning of the whole process of data transformation. You select either whole records or parts of several records from the source systems.
- **Splitting/joining:** Sometimes (uncommonly), you will be splitting the selected parts even further during data transformation. Joining of parts selected from many source systems is more widespread in the data warehouse environment.
- **Conversion:** It includes conversions of single fields to make the fields usable and understandable to the users.
- **Summarization:** Sometimes you may find that it is not feasible to keep data at the lowest level of detail in your data warehouse. So, in this case, the data transformation function includes summarization.
- **Enrichment:** It requires the use of one or more fields from the same input record to create a better view of the data for the data warehouse.

QUERIES

Denormalization:

1. **Horizontal split:**
   Create table new as
   select * from hr.countries
   where region_id = 3;
2. **Vertical split:**
   Create table new as
   select job_id from hr.job_history;
3. **Pre joining:**
   Create table new as
   Select l.*, c.* from hr.locations l, hr.countries c
   where l.country_id = c.country_id;
4. **Adding redundant column:**
   Create table new as
   select d.*, l.street_address from hr.departments d, hr.locations l
   where d.location_id = l.location_id;
5. **Collapsing (many-to-many):**
   Create table new as
   select jh.end_date, jh.department_id, e.* from hr.job_history jh, hr.employees e
   where jh.job_id = e.job_id;
6. **Derived attributes:**
   Create table new as
   Select CURRENT_DATE – hire_date “Working days” from hr.employees;

Indexing

Inverted index:

An inverted index is an optimized structure that is built primarily for retrieval, with update being only a secondary consideration. The basic structure inverts the text so that instead of the view obtained from scanning documents where a document is found and then its terms are written, an index is built that maps terms to documents. Instead of listing each document once (and each term repeated for each document that contains the term), an inverted index lists each term in the collection only once and then shows a list of all the documents that contain the given term.

Cluster index:

Clustered indexes are indexes whose order of the rows in the data pages corresponds to the order of the rows in the index. This order is why only one clustered index can exist in any table.

Dense vs sparse indexing:

DENSE INDEX	SPARSE INDEX
An index record appears for every search key value in file.	An index record are created only for some of the records.
Indicies are faster	Indicies are slower
Indicies requires more space	Indicies requries less space
Indicies impose more maintainance for insertions and deletions.	Indicies impose less maintainance for insertions and deletions

Development Methodologies

Data-Driven Methodologies: Bill Inmon, the founder of data warehousing says that requirements are the last thing to be considered in the development lifecycle. Requirements are understood AFTER the data warehouse has been populated with data and results of queries have been analyzed by the end users. Consequently company goals and user requirements are not reflected at all in the first cycle, and are integrated in the second cycle.

Goal-Driven Methodologies: In order to derive the initial data warehouse structure, a four-stage approach based on the SOM (Semantic Object Model) process modeling technique has been presented.

- The first stage determines goals and services the company provides to its customers.
- In the second step, the business process is analyzed by applying the SOM.
- In third step, sequences of transactions are transformed into sequences of existing dependencies.
- The last step identifies measures and dimensions.

Kimball also proposes a four-step approach where he starts to choose a business process, takes the grain of the process, and chooses dimensions and facts. He defines a business process as a major operational process in the organization that is supported by some kind of legacy system (or systems).

User-Driven Methodologies: The methodology assumes that the company goal is the same for everyone and the entire company will therefore be pursuing the same direction. It is proposed to set up first

prototype based on the needs of the business. Business people define goals and gather, priorities as well as define business questions supporting these goals. Afterwards the business questions are prioritized and the most important business questions are defined.

Kimball's Goal-Driven Approach

Kimball proposes a four-step approach where he starts to choose a business process, takes the grain of the process, and chooses dimensions and facts. He defines a business process as a major operational process in the organization that is supported by some kind of legacy system (or systems).

Lifecycle

Lifecycle begins with project planning during which we assess the organization's readiness for a data warehouse initiative, establish the preliminary scope and justification, obtain resources, and launch the project.

The second major task focuses on business requirements definition. Data warehouse designers must understand the needs of the business and translate them into design considerations. Business users and their requirements have an impact on almost every design and implementation decision made during the course of a warehouse project.

The top track deals with technology. Technical architecture design establishes the overall framework to support the integration of multiple technologies.

The middle track spreading out from business requirements definition focuses on data. We begin by translating the requirements into a dimensional model which is then transformed into a physical structure. **Last but not least, data staging Extract -Transform-Load (ETL) processes are designed and developed.**

The final set of tasks spawned by the business requirements definition is the design and development of analytic applications. The data warehouse project isn't done when we deliver data. Analytic applications will satisfy a large percentage of the analytic needs of business users.

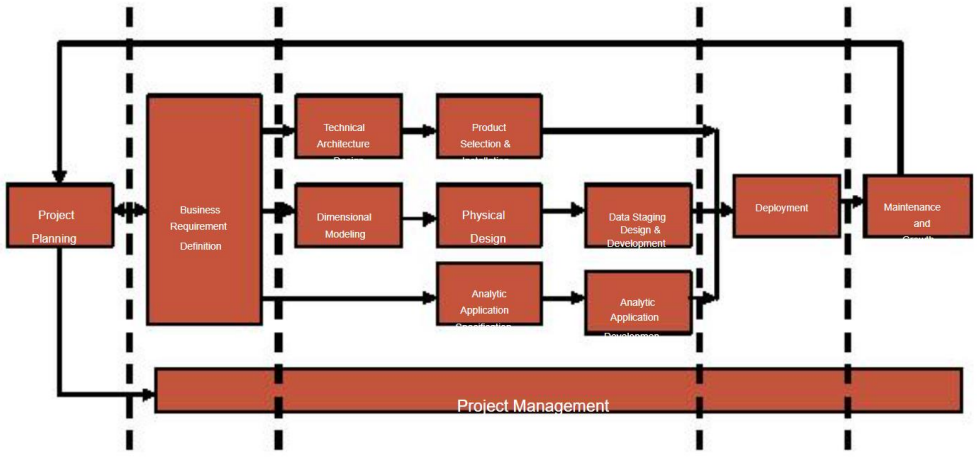

Figure -33.1: Business Dimensional Lifecycle (Kimball's Approach)

Data cleansing

Data cleansing: It is crucial to the process of loading the data warehouse that you not just rearrange the grouping of source data and deliver it to a destination database, but you also ensure the quality of that data. Data cleansing is vitally important.

The true scope of a data cleansing project is enormous. Much of production data is dirty. By "dirty," it means that it does not conform to proper domain definitions or "make sense." The age -old adage "garbage in, garbage out" still applies, and you can do nothing about it but analyze and correct the corporate data. What is noise in one domain may be information in another.

Usually the process of data cleansing cannot be performed without the involvement of a domain expert because the detection and correction of anomalies requires detailed domain knowledge. Data cleansing is therefore described as semi-automatic but it should be as automatic as possible because of the large amount of data that usually is be processed and the time required for an expert to cleanse it manually.

The original aim of data cleansing was to eliminate duplicates in a data collection.

3 classes of anomalies

Syntactically Dirty Data:

- **Lexical errors:** For example, assume the data to be stored in table form has each row representing a tuple and each column an attribute. If we expect the table to have five columns because each tuple has five attributes but some or all of the rows contain only four columns then the actual structure of the data does not conform to the specified format.
- **Irregularities:** Irregularities are concerned with the non-uniform use of values, units and abbreviations. This can happen, for example, if we use different currencies to specify an employee's salary. This happens if the currency is not explicitly listed with each value, and is assumed to be uniform. Annual salary 20,000 means $20,000 or Rs. 20,000?

Semantically dirty data:

- **Integrity constraint violations** describe tuples (or sets of tuples) that do not satisfy one or more of the integrity constraints. Integrity constraints are used to describe our understanding of the mini-world. Each constraint is a rule representing knowledge about the domain and the values allowed for representing certain facts.
- **Contradictions** are values within one tuple or between tuples that violate some kind of dependency between the values. An example for the first case could be a contradiction between the attribute AGE and DATE_OF_BIRTH for a tuple representing persons.
- **Duplicates** are two or more tuples representing the same entity from the mini -world. The values of these tuples do not need to be completely identical.

Coverage Problems:

- **Missing values:** Result of omissions while collecting the data. A constraint violation if we have null values for attributes where NOT NULL constraint exists.
- **Missing tuples:** Result from omissions of complete entities existent in the mini-world that are not represented by tuples in the data collection.

Key based classification problems

Primary key problems:

- Records may have the same primary key but might have different data.
- The same entity might occur with different primary keys.
- Data may have a primary key in one system but not in another.
- Primary Keys may be intended to be the same but might occur in different formats.

Non-Primary key problems:

- Data may be encoded differently in different sources.
- There are often multiple ways to represent the same piece of information.
- Sources might contain invalid data.
- Two fields may contain different data but have the same name.
- Required fields may be left blank.
- Data may be erroneous or inconsistent.
- Data may contain null values.

Keys in data warehouse:

In a data warehouse, a key is a field or set of fields that uniquely identify each record in a table. There are several types of keys that can be used in a data warehouse, including:

- Primary Key: A primary key is a unique identifier for each record in a table. It cannot contain null values and must be unique for every record.
- Foreign Key: A foreign key is a field in a table that refers to the primary key of another table. It is used to establish a relationship between two tables.
- Surrogate Key: A surrogate key is a unique identifier for each record in a table that is used instead of the natural key.
- Candidate Key: A candidate key is a field or set of fields that could serve as the primary key for a table, but a different key has been chosen as the primary key.
- Alternate Key: An alternate key is a candidate key that has been selected to be used as the primary key, but it is not the primary key.

These keys play a crucial role in maintaining the integrity and consistency of data in a data warehouse.

Automatic Data Cleansing

1. **Statistical Methods:** Identifying outlier fields and records using the values of mean, standard deviation, range, etc.
2. **Pattern-based:** Identify outlier fields and records that do not conform to existing patterns in the data. Combined techniques (partitioning, classification, and clustering) are used to identify patterns that apply to most records.
3. **Clustering:** Identify outlier records using clustering based on Euclidian (or other) distance. Existing clustering algorithms provide little support for identifying outliers. The main drawback of this method is computational time.
4. **Association rules:** The power of association rules is that they can deal with data of different types.

Data Quality

What Is Data Quality?

There are two significant definitions of data quality. One is its intrinsic quality, and the other is its realistic quality. **Intrinsic data quality** is the correctness or accuracy of data. **Realistic data quality** is the value that correct data has in supporting the work of the business or enterprise. Data that does not help or enable the enterprise to accomplish its mission has no quality, no matter how accurate it is.

Data quality techniques

Data quality assessment techniques

Simple Ratios:

The simple ratio measures the ratio of desired outcomes to total outcomes.

- **Free-of-error:** The free-of-error dimension represents data correctness. If one is counting the data units in error, the metric is defined as the number of data units in error divided by the total number of data units subtracted from 1.
- **Completeness:** At the most abstract level, one can define the concept of schema completeness, which is the degree to which entities and attributes are not missing from the schema. At the data level, one can define column completeness as a function of the missing values in a column of a table. A third type is called population completeness.
  Each of the three types (schema completeness, column completeness, and population completeness) can be measured by taking the ratio of the number of incomplete items to the total number of items and subtracting from 1.
- **Consistency:** A metric measuring consistency is the ratio of violations of a specific consistency type to the total number of consistency checks subtracted from one.

Min-Max Operation:

To handle dimensions that require the aggregation of multiple data quality indicators (variables), the minimum or maximum operation can be applied.

- **Believability:** Believability is the extent to which data is regarded as true and credible. Each of the variables is rated on a scale from 0 to 1, and overall believability is then assigned as the minimum value of all. Assume the believability of the data source is rated as 0.6; believability against a common standard is 0.8; and believability based on experience is 0.7. The overall believability rating is then 0.6. *Min {0.8, 0.7, 0.6) = 0.6*
- **Appropriate amount of data:** A working definition of the appropriate amount of data should reflect the data quantity being neither too little nor too much. It is the ratio of the number of data units provided to the number of data units needed, and the ratio of the number of data units needed to the number of data units provided. *Min {Dp/Dn , Dn/Dp}*
- **Timeliness** reflects how up-to-date the data is with respect to the task it's used for. A general metric to measure timeliness is to measure the maximum of one of two terms: 0 and one minus the ratio of currency to volatility i.e. *Max (0, 1-C/V).*
  Here, currency (C) is defined as the age (A) plus the delivery time (Dt) minus the input time (It)
  C = A + Dt - It.

- **Accessibility**: The metric emphasizes the time aspect of accessibility and is defined as the maximum value of two terms: 0 or one minus the time interval from request by user to delivery to user divided by the time interval from request by user to the point at which data is no longer useful. *Max {0, 1- Trd/Tru}*

Data Quality Validation techniques

- Referential integrity
- Attribute domain
- Using data quality rules
- Data histograming

How to improve data quality

1. **Process Improvement:** Improve the functional processes used to create, manage, access, and use data.
2. **System Improvement:** Software, hardware, and telecommunication changes can improve data quality. For example, security software can minimize damage done by malicious updates to databases by unauthorized users. Hardware improvements may make batch loads faster.
3. **Policy and Procedure Improvement:** Resolve conflicts in existing policies and procedures and institutionalize behaviors that promote good data quality.
4. **Data Design Improvement:** Improve the overall data design and use data standards. Adding primary key constraints, indexes, and referential integrity constraints etc. can improve database design.

KDD Process

KDD refers to the overall process of discovering useful knowledge from data.

- **Selection:** Acting upon a database of compiled data, the targeted data is determined and variables that will be used to evaluate for knowledge discovery are determined.
- **Pre-processing:** This stage is all about improving the data by incorporating the concept of data cleaning. Predictive models for unreliable data are established in order to predict similarly faulty, missing, attributional mismatched data, then working it out of future processes.
- **Transformation:** This phase concentrates on converting the pre-processed data to the fully utilizable kind. Here the information is organized and sorted, often unified into a single type.
- **Data Mining:** Data mining state is focused on sifting through the transformed data to seek out patterns of interest. These patterns are graphed, trended, and charted in the form particularly helpful to the process the KDD is being conducted for. The method incorporated in this phase involves grouping, clustering, and regression, with the chosen one (or more) dependent on the outcome expected and desired from the process.
- **Interpretation/Evaluation:** The final phase is one during which the data is handed off for interpretation and documentation. At this point, the data has been cleaned, converted, picked apart based on relevant attributes, and framed into visual representations to help humans better evaluate the curated output.

Past Paper

Q1. The Entity-Relationship (ER) model is a common approach for modeling data in a relational database, but it may not be the best choice for a data warehouse for several reasons:

1. Complexity: ER models can become very complex and difficult to understand, especially for large data warehouses that have many entities and relationships.
2. Redundant data: ER models often result in redundant data, which can lead to data inconsistencies and make it difficult to keep the data warehouse up to date.
3. Performance: ER models may not be optimized for querying and analysis, which are the primary functions of a data warehouse. The complex relationships and normalization in an ER model can make queries slow and resource-intensive.
4. Lack of focus on business users: ER models are designed to be generic, but a data warehouse is typically used for specific business purposes, such as sales analysis or customer behavior analysis. An ER model may not provide the level of detail or the organization that is needed for these purposes.

For these reasons, many data warehousing experts prefer dimensional modeling approaches, such as the Kimball approach, for designing data warehouses. Dimensional models are optimized for querying and analysis, and they provide a clear and intuitive representation of the data that is tailored to the needs of business users.

Q2. <u>Conventional vs special indexing</u>

Difference	Conventional Indexing	Special Indexing
Purpose	General-purpose indexing method	Designed for specific types of queries or data structures
Data Structure	Uses a balanced tree data structure	Uses bitmap representation or hash function
Search Performance	Fast search performance for a range of queries, including exact match and range queries	Fast search performance for exact match queries, but can be slower for range queries
Example	B-tree indexing	Bitmap indexing, Hash indexing
Space Utilization	Requires more disk space compared to special indexing methods	Requires less disk space compared to conventional indexing methods
Update Performance	Typically slower update performance compared to special indexing methods	Faster update performance compared to conventional indexing methods
Query Types	Suitable for a wide range of query types	Suitable for specific types of queries such as exact match queries
Query Complexity	Supports complex queries	Typically limited to simple queries
Scalability	Scalable to large datasets	Can struggle with very large datasets

Bitmap index:

A Bitmap index is a type of special index that uses a bitmap representation of the index values. In this representation, each bit in the bitmap corresponds to a specific row in the database table and its value represents the presence or absence of a certain index value in that row.

What characteristics a query should have for a maximum performance of bitmapped index?

For bitmap index to achieve maximum performance, a query should have the following characteristics:

- The WHERE clause contains multiple predicates on low-cardinality columns.
- The individual predicates on these low-cardinality columns select a large number of rows.
- Bitmapped indexes have been created on some or all of these low-cardinality columns.
- The tables being queried contain many rows.
- Bitmapped indexes are optimized for exact match queries, so they perform best when searching for exact value in the index.
- Low selectivity meaning a small percentage of rows in the table matches the search criteria.

Q5. Explain Kimball and data vault 2.0 approaches, which is useful for big data warehouse?

Kimball and Data Vault 2.0 are two different methodologies for designing and building data warehouses.

The Kimball methodology was first introduced in the 1990s and is widely used for traditional data warehousing. It emphasizes a dimensional modeling approach, where data is organized into fact tables and dimension tables to represent business processes and dimensions, respectively. The fact tables contain the measures or facts of the business processes, such as sales or inventory levels, while the dimension tables contain the descriptive information about the facts, such as the date, product, or location. The Kimball methodology is based on a star or snowflake schema, where the fact tables are at the center and the dimension tables are connected to them.

Data Vault 2.0, on the other hand, is a more recent approach to data warehousing that was developed to address the challenges posed by big data. It is a hybrid methodology that combines the best features of dimensional modeling and entity-relationship modeling. Data Vault 2.0 uses a centralized hub-and-spoke model to store data in a flexible, scalable, and extensible manner. The hub-and-spoke model consists of three main components: the hub, which represents a unique identifier for a business object, the spoke, which contains the descriptive information about the business object, and the link, which represents the relationships between the hubs and the spokes.

In terms of big data warehousing, Data Vault 2.0 is a better choice because of its ability to handle large and complex data volumes, and its flexibility and scalability in handling the changing and growing requirements of big data. The centralized hub-and-spoke model allows for the easy addition of new data sources and the ability to handle complex relationships between data elements, making it well-suited for big data environments.

In conclusion, both Kimball and Data Vault 2.0 have their strengths and weaknesses, and the choice between them will depend on the specific requirements of the data warehousing project. For traditional data warehousing projects, the Kimball methodology may be a good choice, while for big data warehousing projects, Data Vault 2.0 is a more suitable approach.

Q6 (a) Data mining from business perspective with example.

Data mining is the process of discovering meaningful insights and patterns in large amounts of data. From a business perspective, data mining is used to improve decision-making by identifying hidden trends, relationships, and patterns in data that can inform business strategy and decision-making.

For example, a retail company may use data mining to analyze customer purchase patterns to determine which products are selling well and which are not. The company could then use this information to adjust its product offerings and marketing strategies to improve sales. By analyzing large amounts of sales data, the company can identify the most profitable products and target those to customers who are most likely to buy them, resulting in increased profits.

Another example is a bank using data mining to detect fraud. By analyzing large amounts of transaction data, the bank can identify patterns of behavior that indicate a high probability of fraud. This information can then be used to take preventative measures to stop the fraud before it happens, saving the bank time and money.

In conclusion, data mining is a powerful tool for businesses to gain valuable insights into their operations, customers, and products. By discovering patterns in data, businesses can make informed decisions that improve their bottom line and increase their competitiveness in the market.