Topic 8: Data Warehouse Concepts and Design

1. OLAP vs OLTP

OLTP (Online Transaction Processing)

Purpose: Day-to-day operations

Characteristics:

- **Fast, concurrent** data access
- **Transaction processing** and concurrency control
- Focus on **data consistency** for online updates
- **Detailed data** (current state)
- **No historical data**
- **Highly normalized** schemas
- **Poor performance** on complex queries (many joins, aggregations)

Example Query:

```
-- Simple, specific query
SELECT * FROM Orders
WHERE CustomerID = 123 AND Status = 'Pending';
```

OLAP (Online Analytical Processing)

Purpose: Data analysis and decision support

Characteristics:

- **Heavy query load**
- Focus on **analytical queries**
- **Historical data** included
- **Denormalized** schemas (star, snowflake)
- **Optimized** for aggregation and complex queries
- **Read-mostly** operations

Example Query:

```
-- Complex analytical query
SELECT Product, Region, Year, SUM(Sales)
FROM FactSales
JOIN DimProduct, DimRegion, DimTime
GROUP BY Product, Region, Year;
```

Key Differences Table

Aspect	OLTP	OLAP
Purpose	Operations	Analysis
Data	Current, detailed	Historical, aggregated
Schema	Normalized	Denormalized
Queries	Simple, fast	Complex, aggregated
Users	Many concurrent	Fewer analytical
Updates	Frequent	Batch, periodic

2. The Multidimensional Model

Core Concepts

Data Cube

- **n-dimensional space** for data analysis
- Composed of **dimensions** and **facts**

Dimensions

- **Perspectives** for analyzing data
- Example: Product, Time, Customer
- Have **attributes** describing them
- Organized in **hierarchies**

Facts

- **Measures** at intersection of dimensions
- **Numeric values** for analysis
- Examples: Sales amount, quantity

Measures

- **Numeric values** in facts
- Types:
    1. **Additive**: Can sum across all dimensions (e.g., SalesAmount)
    2. **Semi-additive**: Sum across some dimensions (e.g., Inventory - not Time)
    3. **Non-additive**: Cannot sum (e.g., Ratios, Percentages)

Example: Sales Data Cube

```
Dimensions:
- Product (Category → Subcategory → Product)
- Time (Year → Quarter → Month)
- Customer (Country → City)
```

```
Facts:
- Measure: Quantity sold

Cell: Quantity[Seafood, Q1, Paris] = 150
```

Data Granularity

- **Level of detail** for measures
- Example: Monthly vs. Daily sales
- Coarser granularity = Higher aggregation
- Finer granularity = More detail

3. Dimension Hierarchies

Hierarchy Concepts

- **Sequence of mappings**: Lower-level → Higher-level
- **Child level**: Lower, more detailed
- **Parent level**: Higher, more aggregated
- **Leaf level**: Finest granularity
- **Root level**: Coarsest granularity
- **Members**: Instances at each level

Types of Hierarchies

1. Balanced Hierarchies

Schema Level:

- **One path** only
- **All relationships**: Many-to-one, mandatory

Instance Level:

- Forms **balanced tree**
- All branches **same length**
- Example: Time (Day → Month → Quarter → Year)

2. Unbalanced Hierarchies

Schema Level:

- **One path**
- Some relationships **optional**

Instance Level:

- Forms **unbalanced tree**
- Different branch lengths
- Example: Organizational chart (some levels skipped)

3. Recursive Hierarchies

- **Special case** of unbalanced
- **Same level** linked by parent-child
- Used when levels have **same semantics**
- Example: Employee hierarchy (Employee supervises Employee)

4. Generalized Hierarchies

Schema Level:

- **Multiple exclusive paths**
- Share at least **leaf level**

Instance Level:

- Each member belongs to **only one path**
- Example: Customer (Retail path vs. Corporate path)

5. Noncovering (Ragged) Hierarchies

- **Special case** of generalized
- Obtained by **skipping intermediate levels**
- **Different path lengths** from leaves to same parent
- Example: Geography (City → State → Country, but some cities directly to Country)

Hierarchy Example: Time

```
Day → Month → Quarter → Year
01 → Jan → Q1 → 2024
02 → Jan → Q1 → 2024
...
31 → Dec → Q4 → 2024
```

4. OLAP Operations

Key Operations

1. Roll-up (Drill-up)

- **Aggregate** along dimension hierarchy
- Move to **coarser** granularity
- Example: Month → Quarter → Year

```
ROLLUP(Sales, Time → Quarter, SUM(Quantity))
```

2. Drill-down

- Move from **general to detailed**
- **Opposite** of roll-up
- Example: Year → Quarter → Month

```
DRILLDOWN(Sales, Time → Month)
```

3. Slice

- **Fix** dimension at specific value
- **Reduces** dimensionality by one
- Example: Select only Q1 sales

```
SLICE(Sales, Time.Quarter = 'Q1')
-- Result: 2D cube (Product × Customer)
```

4. Dice

- Select **sub-cube** using conditions
- **Multiple dimensions** filtered
- Example: Q1-Q2, Paris-Lyon

```
DICE(Sales,
     (Customer.City = 'Paris' OR City = 'Lyon') AND
     (Time.Quarter = 'Q1' OR Quarter = 'Q2'))
```

5. Pivot (Rotate)

- **Rotate** cube axes
- **Reorient** visualization
- No data change

```
PIVOT(Sales, Time → X, Customer → Y, Product → Z)
```

6. Sort

- **Order** dimension members
- By name or measure value

```
SORT(Sales, Product, NAME ASC)
```

5. Data Warehouse Architecture

Architecture Tiers

1. Back-end Tier

Components:

- **ETL Tools**: Extract, Transform, Load
    - Extract from operational databases
    - Transform data (clean, integrate)
    - Load into warehouse
- **Data Staging Area**:
    - Intermediate database
    - Integration and transformation
    - Before loading to warehouse

2. Data Warehouse Tier

Components:

- **Enterprise Data Warehouse**: Central repository
- **Data Marts**: Subject-specific subsets
- **Metadata Repository**: Information about warehouse and contents

3. OLAP Tier

Components:

- **OLAP Server**: Provides multidimensional view
- Works regardless of physical storage
- Supports OLAP operations

4. Front-end Tier

Components:

- **Client Tools**:
    - OLAP tools
    - Reporting tools
    - Statistical tools
    - Data mining tools
- **Data analysis and visualization**

OLAP Technologies

ROLAP (Relational OLAP)

- Stores data in **relational databases**
- SQL extensions for OLAP
- **Advantages**: Scalability, uses existing RDBMS
- **Disadvantages**: Slower than MOLAP

MOLAP (Multidimensional OLAP)

- Stores in **specialized structures** (arrays)
- **Advantages**: Fast query, pre-aggregated data
- **Disadvantages**: Limited scalability, requires specialized storage

HOLAP (Hybrid OLAP)

- **Combines** ROLAP and MOLAP
- **Detailed data**: Relational storage
- **Aggregations**: Multidimensional storage
- **Best of both worlds**

6. Logical Data Warehouse Design

Star Schema

Structure:

- **One fact table** (center)
- **Dimension tables** (points of star)
- Dimension tables **denormalized**

Advantages:

- Simple structure
- Fast queries
- Easy to understand

Disadvantages:

- Data redundancy in dimensions
- Larger dimension tables

Example:

```
          DimProduct
               |
DimCustomer - FactSales - DimTime
               |
          DimRegion
```

Snowflake Schema

Structure:

- Fact table (center)
- **Normalized** dimension tables
- Dimension hierarchies = separate tables

Advantages:

- Reduced redundancy
- Smaller dimension tables
- Better for complex hierarchies

Disadvantages:

- More complex structure
- More joins required
- Slower queries than star

Example:

```
DimCity - DimState - DimCountry
                          |
                     FactSales
```

Starflake Schema

- **Hybrid** of star and snowflake
- **Some dimensions** normalized
- **Others** denormalized
- Balance between simplicity and storage

Constellation Schema

- **Multiple fact tables**
- **Share dimension tables**
- Common dimensions across facts
- Example: Sales and Inventory facts share Product, Time dimensions

7. Conceptual DW Design (MultiDim Model)

Key Components

Level

- **Entity type** in dimension hierarchy
- Has attributes (key, descriptive)

Dimension

- One or more **hierarchies**
- Provides analysis perspective

Fact

- Relates **measures** to **leaf levels** in dimensions
- Can have different cardinalities with dimensions

Cardinality

- **Minimum/maximum** members in level
- Related to members in another level
- Example: 1..n (one-to-many)

MultiDim Notation Elements

- **Rectangle**: Level
- **Arrow**: Parent-child relationship
- **Numbers**: Cardinality (min..max)
- **Diamond**: Fact
- **Circle**: Measure

8. Physical DW Design

Techniques

1. Materialized Views

Definition: Physically stored query results

Advantages:

- **Faster queries**: No recomputation
- **Pre-computed joins**: Eliminates join overhead

Types:

- **Brute Force**: Explicit access to view
- **Transparent Query Rewrite**: Optimizer uses view automatically

Maintenance:

- **Incremental**: Update only changes
- **Complete**: Rebuild entire view

Example:

```
CREATE MATERIALIZED VIEW MV_ORDERS
REFRESH ON COMMIT
ENABLE QUERY REWRITE
AS
SELECT O_ORDERKEY, O_CUSTKEY, SUM(O_TOTALPRICE)
FROM ORDERS
WHERE O_ORDERDATE > '2020-01-01'
GROUP BY O_ORDERKEY, O_CUSTKEY;
```

2. Indexes

B-tree Index:

- **Balanced tree** structure
- Good for **range queries**
- Supports **equality** and **range** predicates

Bitmap Index:

- **Bit vector** for each distinct value
- Excellent for **low-cardinality** columns
- **Efficient** for complex AND/OR queries
- Example: Gender (M/F), Status (Active/Inactive)

Bitmap Operations:

```
Products: PiecesPerUnit BETWEEN 45 AND 55
          AND UnitPrice BETWEEN 100 AND 200

1. Get bitmap for 45 ≤ PPU ≤ 55
2. Get bitmap for 100 ≤ UP ≤ 200
3. AND bitmaps
4. Return matching records
```

Index Requirements for DW:

- **Symmetric partial match**: All dimensions indexed
- **Multiple aggregation levels**: Index summary tables
- **Efficient batch update**: Fast refresh
- **Sparse data handling**: Handle empty cells

3. Partitioning

Vertical Partitioning:

- Split **attributes** into groups
- Store groups independently
- Example: Frequently vs. rarely used columns

Horizontal Partitioning:

- Divide table into **smaller tables**
- Same structure, different rows
- Example: Partition by date range

Partitioning Strategies:

1. **Range Partitioning**:
    - Based on **value ranges**
    - Example: Sales by month

```
Jan-2024, Feb-2024, Mar-2024 partitions
```

2. **Hash Partitioning**:
    - Use **hashing algorithm**
    - **Uniform distribution**
    - Good for **non-time** data
3. **List Partitioning**:
    - Specify **list of values**
    - Example: Regions (Asia, Europe, Americas)
4. **Composite Partitioning**:
    - **Combine** strategies
    - Example: Range by month, hash within month

Benefits:

- **Partition pruning**: Access only needed partitions
- **Improved joins**: Partition-wise joins
- **Easier maintenance**: Manage partitions independently

Star Query Evaluation

Star Query:

- Joins fact table with dimension tables
- Selection conditions on dimensions
- Aggregation of results

Example:

```
SELECT ProductName, CustomerName, SUM(SalesAmount)
FROM Sales S, Customer C, Product P
WHERE S.CustomerKey = C.CustomerKey
```

```
  AND S.ProductKey = P.ProductKey
  AND P.Discontinued = 'Yes'
GROUP BY C.CustomerName, P.ProductName;
```

Evaluation Steps:

1. **Evaluate joins**: Fact with dimensions
2. **Apply selections**: Filter on dimensions
3. **Aggregate**: Group and sum results

With Bitmap Indexes:

1. Get bitmap for selection condition
2. Identify matching fact records
3. Use B+-tree to get dimension values
4. Aggregate results

Time Dimension

Characteristics:

- Present in **almost all** data warehouses
- **Historical database** aspect
- Stored as **explicit attributes**

Granularity:

- Varies by use case
- Example: Day, Week, Month, Quarter, Year

Time Table Attributes:

- Date (primary key)
- Day, Month, Quarter, Year
- DayOfWeek, WeekendFlag
- Holiday indicators
- Fiscal periods

Benefits:

- **Easy filtering**: `WHERE WeekendFlag = TRUE`
- **No computation**: Pre-calculated attributes
- **Consistent**: Standardized across queries

Key Points for Exam

Critical Concepts

1. **OLAP vs OLTP**: Analysis vs Operations
2. **Multidimensional Model**: Dimensions + Facts + Measures

3. **Data Cube**: n-dimensional data space
4. **Hierarchies**: Levels of aggregation
5. **Star Schema**: Central fact, denormalized dimensions

OLAP Operations (Must Know)

- **Roll-up**: Coarser granularity
- **Drill-down**: Finer granularity
- **Slice**: Fix one dimension
- **Dice**: Multi-dimensional filter
- **Pivot**: Rotate view

Schema Types

1. **Star**: Simple, fast, redundant
2. **Snowflake**: Normalized, complex, less redundant
3. **Starflake**: Hybrid approach
4. **Constellation**: Multiple facts, shared dimensions

Hierarchy Types

1. **Balanced**: Equal path lengths
2. **Unbalanced**: Optional levels
3. **Recursive**: Level references itself
4. **Generalized**: Multiple exclusive paths
5. **Noncovering**: Skip intermediate levels

Physical Design Techniques

1. **Materialized Views**: Pre-computed results
2. **Indexes**: B*-tree (range), Bitmap (low cardinality)
3. **Partitioning**: Horizontal (rows), Vertical (columns)

OLAP Technologies

- **ROLAP**: Relational storage, scalable
- **MOLAP**: Array storage, fast
- **HOLAP**: Hybrid approach

Important Measures

- **Additive**: Sum across all (SalesAmount)
- **Semi-additive**: Sum across some (Inventory)
- **Non-additive**: Cannot sum (Ratios)

Partitioning Strategies

- **Range**: Value ranges (dates)
- **Hash**: Uniform distribution
- **List**: Explicit values

- **Composite**: Combined approaches