**Chapter 1: The Database Environment**

1. **Definition of data, metadata, and database**

Data: stored representations of meaningful objects and events

Metadata: data that describes the properties and context of user data

Database: organized collection of logically related data

1. **Issues with file processing systems**

Dis: Program-Data Dependence, Duplication of Data, Limited Data Sharing, Lengthy Development Times,Excessive Program Maintenance

1. **Definition of a database management system**

A software system that is used to create, maintain, and provide controlled access to user databases

1. **Advantages and cost/risk of the database approach**

Advantages:Program-data independence, Planned data redundancy, Improved data consistency, Improved data sharing, Increased application development productivity, Enforcement of standards, Improved data quality, Improved data accessibility and responsiveness, Reduced program maintenance, Improved decision support

cost/risk: Specialized personnel, Installation and management cost and complexity, Conversion costs, Need for explicit backup and recovery, Organizational conflict

1. **Elements of the database approach**

Data models, Database management system, Use of Internet Technology, Database Applications

1. **One-to-many/many-to-many relationships**
2. **Components of the database environment**

CASE Tools, Repository,Database Management System (DBMS), Database, Application Programs, User Interface, Administrators, System Developers, End Users

1. **Database applications**

Personal databases, Work-group databases, Departmental/divisional databases, Enterprise database

1. **Enterprise data model**

Enterprise Resource Planning, Data Warehouse, Big Data and Business Analytic

**Chapter 2: ER Diagram**

1. **Definition of business rules and good data names**

business rules:Statements that define or constrain some aspect of the business

good data names: Related to business, not technical, characteristics, Meaningful and self-documenting , Unique, Readable, Composed of words from an approved list, Repeatable

1. **The three basic constructs for ER models**

Entities, Relationships, Attribute

1. **How to choose good entities and attributes and how to represent relationships correctly**

Entity:

SHOULD BE: An object that will have many instances in the database, An object that will be composed of multiple attributes, An object that we are trying to model SHOULD NOT BE: A user of the database system , An output of the database system

Attribute: Relation:

1. **Different types of attributes**

Required versus Optional Attributes, Simple versus Composite Attribute, Single-valued versus Multivalued Attribute, Stored versus Derived Attributes, Identifier Attributes

1. **Characteristics of identifiers**

Identifier (Key): An attribute (or combination of attributes) that uniquely identifies individual instances of an entity type

Will not change in value, Will not be null, No intelligent identifiers (e.g., containing locations or people that might change), Substitute new, simple keys for long, composite keys

1. **Degree of relationships: unary, binary, ternary**
2. **Cardinality of relationships: one-to-one, one-to-many, many-to-many**

One-to-One: Each entity in the relationship will have exactly one related entity One-to-Many: An entity on one side of the relationship can have many related entities, but an entity on the other side will have a maximum of one related entity

Many-to-Many: Entities on both sides of the relationship can have many related entities on the other side

1. **Multiple relationships**

Multiple relationships:Entities can be related to one another in more than one way

1. **Strong vs Weak entities, Associative entities**

Strong entity: exists independently of other types of entities, has its own unique identifier, identifier underlined with single-line 

Weak entity: dependent on a strong entity (identifying owner)cannot exist on its own, does not have a unique identifier (only a partial identifier), Partial identifier underlined with double-line, Entity box has double line

Associative Entity: An entity has attributes, A relationship links entities together

1. **Supertype/subtype relationships**

Subtype: A subgrouping of the entities in an entity type that has attributes distinct from those in other subgroupings

Supertype: A generic entity type that has a relationship with one or more subtypes

1. **Entity clusters**

Entity cluster: Set of one or more entity types and associated relationships grouped into a single abstract entity type

1. **Action assertions in ER diagrams**

Action assertions identify corresponding objects that constrain the ability to perform actions on anchor objects

**Chapter 3: Introduction to SQL**

1. **Benefits of a Standardized Relational Language**

Reduced training costs, Productivity, Application portability, Application longevity, Reduced dependence on a single vendor, Cross-system communication

1. **The SQL environment**

Catalog: A set of schemas that constitute the description of a database 

Schema: The structure that contains descriptions of objects created by a user

Data Definition Language (DDL): Commands that define a database, including creating, altering, and dropping tables and establishing constraints 

Data Manipulation Language (DML): Commands that maintain and query a database 

Data Control Language (DCL): Commands that control a database, including administering privileges and committing data

1. **DDL, DML, DCL, and the database development process**

**DDL:**

**CREATE**

Referential integrity: constraint that ensures that foreign key values of a table must match primary key values of a related table in 1:M relationships

**ALTER**

ALTER TABLE table\_name alter\_table\_action; 

ADD [COLUMN] column\_definition 

ALTER [COLUMN] column\_name SET DEFAULT default-value 

ALTER [COLUMN] column\_name DROP DEFAULT 

DROP [COLUMN] column\_name [RESTRICT] [CASCADE] 

ADD table\_constraint

**DROP**

DROP TABLE Customer\_T

**CREATE INDEX**

CREATE INDEX CustomerNameIdx ON Customer\_T(CustomerName)

**DML:**

**INSERT**

INSERT INTO Product\_T (ProductID, ProductDescription) VALUES (1, 'End Table');

INSERT INTO CA\_Customer\_T SELECT \* FROM Customer\_T WHERE CustomerState = 'CA';

**DELETE**

DELETE FROM Customer\_T WHERE CustomerState = 'HI';

**UPDATE**

UPDATE Product\_T SET ProductStandardPrice = 775 WHERE ProductID= 7;

**SELECT**

SELECT FROM WHERE GROUP BY HAVING ORDER BY

1. **Aggregate functions and Boolean operators**

COUNT, MAX, MIN, SUM, AVG, AND, OR, NOT, LIKE, IN(),

%代表任意字符，

1. **Using and defining views; pros/cons of dynamic views**

Advantages of Dynamic Views: Contain most current base table data, Use little storage space Disadvantages of Dynamic Views: Use processing time each time view is referenced, May or may not be directly updateable

1. **Different types of JOIN operations; multiple-table SQL queries**

Join, Natural join (inner join), Outer join, Union join

INNER JOIN, (LEFT) OUTER JOIN

1. **Noncorrelated and correlated subqueries**

Noncorrelated subqueries: Do not depend on data from the outer query, Execute once for the entire outer query 

Correlated subqueries: Make use of data from the outer query, Execute once for each row of the outer query, Can use the EXISTS operator

**Chapter 4: Logical Database Design and Relational Model**

1. **Properties of relations**

Definition: A relation is a named, two-dimensional table of data

Table consists of rows (records) and columns (attribute or field)

All relations are in 1st Normal Form

1. **Definition of candidate keys, primary keys, foreign keys, simple keys, and composite keys.**

candidate keys: A set of attributes that uniquely identifies a tuple in a relationship without redundant attributes

primary keys: unique identifiers of the relation in question

foreign keys: identifiers that enable a dependent relation to refer to its parent relation

simple keys: a single field

composite keys: more than one field

1. **Different types of integrity constraints**

Domain Constraints: Allowable values for an attribute.

Entity Integrity: No primary key attribute may be null. All primary key fields MUST have data

Action Assertions: Business rules.

Referential Integrity: rule states that any foreign key value (on the relation of the many side) MUST match a primary key value in the relation of the one side.

1. **Transforming EER diagrams into relations**
2. **Goal of data normalization**

avoid unnecessary duplication of data

1. **Types of anomalies**

Insertion Anomaly: adding new rows forces user to create duplicate data

Deletion Anomaly: deleting rows may cause a loss of data that would be needed for other future rows

Modification Anomaly: changing data in a row forces changes to other rows because of duplication

1. **Identify functional dependencies in relations; express them using arrows or diagrams**

Functional Dependency: The value of one attribute (the determinant) determines the value of another attribute

1. **Understand the differences between 1NF, 2NF, and 3NF and how to do normalization**

FIRST NORMAL FORM: No multivalued attributes, Every attribute value is atomic

SECOND NORMAL FORM: 1NF PLUS every non-key attribute is fully functionally dependent on the ENTIRE primary key

THIRD NORMAL FORM: 2NF PLUS no transitive dependencies (functional dependencies on non-primary-key attributes)

1. **Issues to watch out in merging relations**

Synonyms: two or more attributes with different names but same meaning

Homonyms: attributes with same name but different meanings

Transitive dependencies: even if relations are in 3NF prior to merging, they may not be after merging

Supertype/subtype relationships: may be hidden prior to merging

**Chapter 6: Data Warehouse**

1. **Purpose of data warehousing**

Support business performance management (business intelligence), Support knowledge discovery and data mining (business analytics)

1. **Characteristics of data warehouses**

Subject oriented, Integrated, Time-variant (time series), Non-volatile, Summarized, Not normalized, Metadata, Relational/multidimensional, Web based, client/server, cloud-based, Real-time/right-time/active...

1. **Data warehouse architectures**

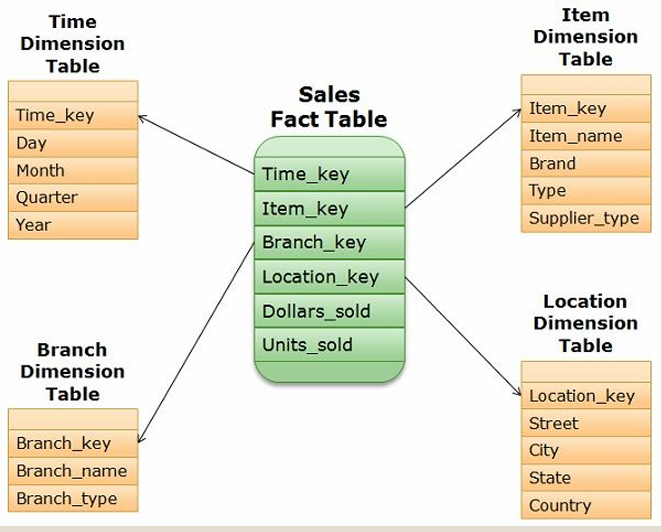
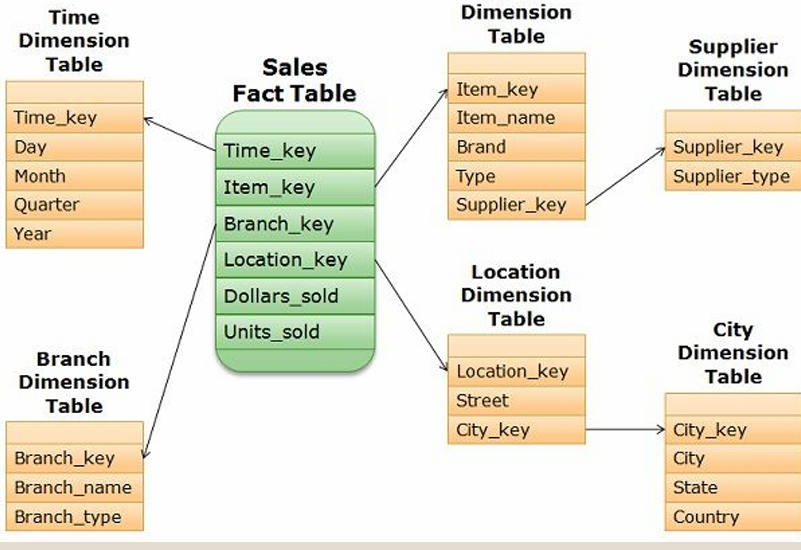
Generic Architecture, Independent Data Mart, Dependent Data Mart and Operational Data Store, Logical Data Mart and Real-Time Data Warehouse, Three-Layer architecture

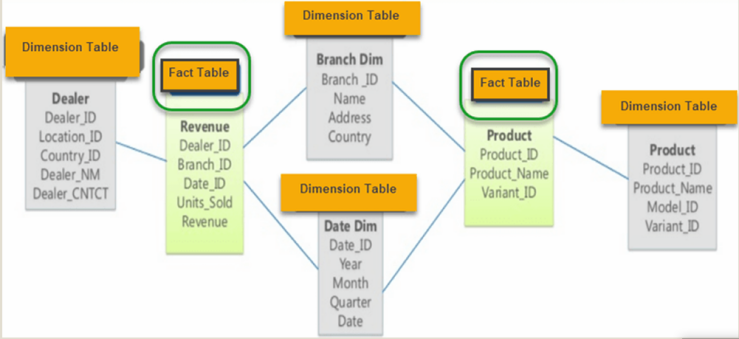
1. **The ETL process**

Capture/Extract, Scrub or data cleansing, Transform, Load and Index

1. **Compare and contrast star, snowflake, and galaxy schemas**

Star: Snowflake:



Galaxy：

1. **Discuss basic OLAP operations**

Slice: a subset of a multidimensional array

Dice: a slice on more than two dimensions

Drill Down/Up: navigating among levels of data ranging from the most summarized (up) to the most detailed (down)

Roll Up: computing all of the data relationships for one or more dimensions

Pivot:used to change the dimensional orientation of a report or an ad hoc query page display

1. **Discuss the success factors for data warehouse implementation**

Identification of data sources and governance, Data quality planning, data model design, ETL tool selection, Establishment of service-level agreements, Data transport, data conversion, End-user support

**Chapter 7: NoSQL Databases**

1. **Issues with RDBMS when used for applications nowadays**

Mismatch with Today’s Workloads: Data: Large and unstructured, Lots of random reads and writes, Sometimes write-heavy, Foreign keys less needed, Joins infrequent, Scale out, not Scale up

1. **Compare the pros and cons of RDBMS and NoSQL databases**

Pro: Elastic Scaling, DBA Specialists, Big Data, Flexible data models, Economics

Con: Support, Maturity, Administration, Lack of Expertise, Analytics and Business Intelligence

1. **Discuss the key features of NoSQL models**

Non-relational, Do not require schema, Data are replicated to multiple nodes and can be partitioned, Horizontal scalable, Cheap, easy to implement (open-source), Massive write performance, Fast key-value access

1. **Discuss the major categories of NoSQL databases**

1.Standard key-value • Example: DynamoDB, Voldermort, Scalaris

2.Document-based • Example: MongoDB, CouchDB

3.Column-based • Example: BigTable, Cassandra, HBase

4.Graph-based • Example: Neo4J, InfoGrid

5.Vector-based • Example: Pinecone, Milvus, Chroma

1. **The CAP Theorem and its implications**

In a distributed system you can satisfy at most 2 out of the 3 guarantees:

1.Consistency: all nodes see same data at any time, or reads return latest written value by any client

2.Availability: the system allows operations all the time, and operations return quickly 3.Partition-tolerance: the system continues to work in spite of network partitions (i.e., network failure)

CAP theorem implies that a system has to choose between consistency and availability

**Chapter 8: MongoDB**

1. **Write simple Javascript statements to perform read, create, update, and insert operations on a MongoDB database**

db.createCollection("CollectionName")

db.collectionName.drop()

db.collectionName.updateMany({},{$set:{new\_field:""}})

db.collectionName.updateMany({},{$unset:{"new\_field": ""}})

db.collectionName.insertOne({ Field1: Value1,  Field2: Value2, Field3: Value3})

db.collectionName.find({Field1: Value1},  {Field1: 1, Field2: 1})

db.collectionName.countDocuments({})

db.collectionName.find({ Field1: { $gt: Value1 } })

db.collectionName.find({$or:[{Field1:Value1},{Field2:Value2}]})

db.collectionName.find( { Field1: /^Value/ } )