INF3703 Study Notes

Chapter 10

- An information system is composed of people, hardware, software, databases, applications and procedures
- Systems analysis is the process that establishes the need for and the scope of an information system
- Systems development – the process of creating an information system
- The performance of an information system depends on DB design and implementation; app design and implementation; and administrative procedures
- DB Design objectives – create complete, normalised, non-redundant and fully integrated conceptual, physical and logical DB models

SDLC Phases (PADIM)

- Planning (initial assessment and feasibility study)
- Analysis (user requirements, existing system evaluation and logical system design)
    - Thorough audit of user requirements
    - Development of data models
- Detailed systems design(detailed system spec)
- Implementation (coding, testing and debugging, installation, fine-tuning)
- Maintenance (Evaluation, Maintenance and Enhancement)
    - Corrective changes – in response to errors
    - Adaptive changes – changing business environment
    - Perfective maintenance –system enhancement

Database Lifecycle (IDITOM)

- Database Initial Study
    - Analyse the company situation (company objectives, operations and structure)
    - Define problems and constraints
    - Define objectives
    - Define scope and boundaries
- Design (most critical phase)
    - **2 views of data – business view and designer's view**

Database Design Flow

a) Conceptual Design (DBMS-independent) - create an abstract DB structure that represents real-world objects
    - **DB analysis and requirements** (user views, output and transaction-processing requirements)
    - **ER modelling and normalisation** (entities, attributes, relations, ER diagrams, table normalisation)
    - **Data model verification** – identify main processes, validate reports, queries, views, integrity, sharing and security

    - **Distributed DB design** – define location of tables, access requirements and fragmentation strategy
  b) DBMS Software selection (costs, features, underlying model, portability and hardware requirements)
  c) Logical design (DBMS-dependent) – Translate conceptual model into definitions for tables, views, etc. Objective – to map the conceptual model into a logical model which can be implemented on an RDBMS.
    - Create logical data model
    - Validate logical data model using normalisation
    - Assign and validate constraints
    - Merge various local models
    - Review logical model with user
  d) Physical Design (Hardware-dependent) – define storage structure and access paths
    - Analyse data and usage
    - Translated logical model relations into tables
    - Define indexes
    - Define user views
    - Estimate data storage requirements
    - Determine DB security for users

- Implementation and Loading (create DB, Create views, relations, assign rights, load DB)
    - Also includes security, performance, integrity, standards; and backup and recovery Oracle encryption feature – Transparent Data Encryption – can decrypt DB columns. Circuit-level gateway blocks all incoming connections to any host but itself

- Testing and Evaluation (testing and fine-tuning, modify physical design, modify logical design, upgrade DBMS)

- Operation
- Maintenance and Evolution
    - Preventative, corrective, adaptive, access permission assignment, DB stats, security audits and periodic summaries and reports

DB Design Strategies (influenced by scope and size of system, company operations, mgmt. style and structure)

- Top-Down – Identify entities/data sets and the elements within the entities – ER Models
- Bottom-Up – Identify the elements and group then into entities/data sets – Normalisation

Centralised vs. Decentralised DB Design

Centralised

- Normally suited for smaller companies or small databases where 1 DBA designer can define the problem, create the conceptual model, verify conceptual model, etc.
- Illustration – (Conceptual model – (conceptual model verification (user views, system processes, data constraints)), data dictionary)

Decentralised

- When the data component of a system requires many entities with complex relations on which complex operations are performed.
- Teams of designers independently create the conceptual model for a particular component and each conceptual model is merged to form one overall conceptual model
- Illustration – (Data Component – (conceptual models, verification, aggregation, conceptual model, data dictionary)

DBA Roles

- DA (Information Resource Manager) has higher authority than DBA

DBA Managerial Role (Controlling and Planning)

- **End-User Support** – gather user reqs, build end-use confidence, resolve conflicts and problems, find solutions to information needs, ensure quality and integrity of applications and data; and manage the training and support of end users
- Data backup and recovery
- Data distribution and use – Ensure that data is distributed to the right people at the right time
- **Data security, privacy and integrity**
-
- **User Access Management**
    - Define users, groups, roles
    - Manage user access privileges
    - Assign passwords
    - Control physical access
    - Define views
    - DBMS usage monitoring, access control

DBA Managerial Roles include:

- End-User Support
- Policies, Procedures and Standards
- Data security, Privacy and Integrity
- Data Backup and Recovery
- Data Distribution and Use

Technical DBA Roles

- Evaluating, Selecting and Installing the DBMS and utilities
- Designing and Implementation of databases and applications
- Testing and Evaluation of Databases and Applications
- Operate the DBMS, Utilities and Applications
- Training and Supporting Users
- Maintenance of the DBMS, Utilities and Applications

Critical success criteria for an Information Systems Architecture:

- Management commitment
- End-use involvement

- Thorough company situation analysis
- Defined standards
- Training
- Small pilot project

Chapter 11

DB Design stages – Conceptual, Logical and physical design

- Module cohesivity – Must be high. High cohesivity indicates a strong relationship amongst entities
- Module coupling - must be low. Indicates how independent modules are from each other

CODD'S relational DB rules

- Logical independence – if the logical structure of the database changes, the user should not be affected in any away
- Physical independence – If the physical structure of the database changes, the user should not be affected in any way

Conceptual Design Steps

- **Data analysis and requirements** (info needs, users, sources, constitution)
    - Sources of information (developing and gathering use data views, observing current system and interface with the systems design team)
    - Business rules must be easy to understand and widely disseminated
    - Description of operations – a doc that provides precise, detailed, up-to-date and thoroughly reviewed description of business activities
    - Business Rules have the following benefits:
        - Communication tool between designers and users
        - They allow the designer to understand the nature of, role and scope of user data
        - They allow the designer to understand business processes
        - They allow the designer to develop appropriate relationship participation rules and foreign key constraints
- **Entity relationship modelling and normalisation** (standard t be used for documentation must be communicated)
    - Identify, analyse and refine business rules
    - Identify main entities
    - Define relationships amongst entities
    - Define attributes, primary keys and foreign keys for each entity
    - Normalise the entities
    - Complete the initial ER diagram
    - Verify the model with end-users
    - Modify the ER diagram based on user feedback

- **Data Model Verification**
    - Identify the model's central entity
    - Identify each module and it's components
    - Identify each module's transaction requirements
    - Verify all processes against the ER model
    - Make changes as suggested
    - Repeat steps 2 to 5 for all modules
- **Distributed Database Design** (spreading the DB across multiple distributed DBs)

Logical Database Design

- Create the logical model
    - Create relations for strong entities
    - Create relations for weak entities
    - Map multivalued attributes
    - Map binary relations
    - Map ternary relations
    - Map supertype andsubtype relationships
- Validate the model using normalisation
    - Normalise all entities to @least 3NF
- Assign and validate integrity constraints (domain, entity and referential integrity)
- Merge local models constructed for different parts together
- Review logical model with users

Physical Database Design

- **Analyse data volume and database usage** (shown in composite usage or transaction usage map)
    - Identify most frequent and critical transactions
    - Determine which entities will be accessed by critical transactions
- Translate logical relations into tables
- Determine suitable file organisation
    - **Heap Files** – contain randomly ordered records. Used only when a large quantity of data needs to be added to a database for the first time. Slow searches and impractical.
    - **Sequential file organisations** – Records are usually sequenced based on primary key. Insertions are costly, deletions lead to unused space. Searches are very slow
    - **Indexed file organisations** – files that are sorted based on one more fields. Index is created to quickly locate records. Faster searches, access, aggregation. Indexes are logically and physically independent of the data in the associated table
    - **Hashed file organisation** – Uses a hashing algo to map primary key value to a specific record. Disadvantage – the uniqueness of the hash cannot be guaranteed

    - Types of indexes

- Primary index – placed on unique fields/primary key. One primary index per file.
- Secondary index – can be placed on any field in the file that is unordered
- Multi-level index – used when the index becomes large and is split into separate indexes

Indexes can be sparse or dense. Dense indexes are faster.

- Balanced trees (B-Trees) – more efficient in storing ordered data. Left node < parent node, parent node < right node
- Bitmap indexes – usually applied to attributes which are sparse in their given domain. 2 dimensional arrays. Used in the following instances
  - Column has low cardinality
  - The table is not used for data manipulation
  - Specific queries reference a number of low cardinality values
- Join Indexes (Bitmap Join Indexes) – used in data warehousing and applies to columns from 2 or more tables whose value come from the same domain.

- Creating indexes
  - Primary Index - CREATE UNIQUE INDEX <INDEX_NAME> ON <TABLE>(<CLOUMN_NAME>)
  - Secondary Index – CREATE INDEX <INDEX_NAME> ON <TABLE>(<COLUMN_NAME>)
- Define User Views
- Estimate Data Storage Requirements
  - Estimate size of each row by adding the length of each data type
  - Estimate number of rows taking into consideration growth
  - Multiply the size by the estimated number of rows
- Determine database security for users
  - Two sets of privileges (system and object)
  - Systems allows the executing of DDL commands like CREATE TABLE
  - Object allows the executing of DML commands (insert, update)
  - Roles (CREATE ROLE <Role_Name>, GRANT SELECT ON CUSTOMERS TO <Role Name>, GRANT <Role_Name> to <user_to_be_added_to_role>

Chapter 12

- Concurrency control – managing concurrent transactions
- Transaction – logical unit of work that must be entirely completed or entirely aborted
- Consistent database state – a state in which all data integrity constraints are satisfied
- Transactions must satisfy (ACIDS) – Atomicity, Consistency, Isolation, Durability and Serialisation
  - Atomicity – All transactions must be completed, otherwise the transaction will be aborted
  - Consistency – the permanence of the database's consistent state
  - Isolation – Data used during the executing of a transaction cannot be used by a second transaction until the first one is completed
  - Durability – Once transactions are committed, they cannot be undone
  - Serialisability – Ensures that the concurrent execution of several transactions yields consistent results

- Scheduler – special DBMS program that establishes the order in which transactions are executed. Satisfies isolation and serialisation
- A lock guarantees exclusive use of a data item to a current transaction. Controlled by a lock manager.
- Lock levels – database, table, page or field(attribute)
- Database lock – suitable for batch-processing but not for multi-user databases
- Table lock – locks all rows in a table. Causes bottlenecks. Not suitable for multi-user databases
- Page-level lock – an entire diskpage is locked. Most frequent locking option.
- Row-level lock – Less restrictive. Improves availability of data. Adds management overhead.
- Field-level lock – Allows concurrent transactions to access the same row as long as they require access to different fields in the row. Most flexible but rarely implemented due to overhead
- Lock Types
    - Binary – either locked or unlocked. Too restrictive
    - Exclusive lock – Access is reserved for the transaction using the data. No other access (read or write)
    - Shared locks – concurrent transactions are granted read access as long as the transaction is read-only
- 2-Phase locking – define how transactions acquire and relinquish locks. Ensures serialisation but cannot prevent deadlocks.
    - Growing phase – transaction acquires all required locks without unlocking any data. Transaction is locked once all locks have been acquired
    - Shrinking phase – Transaction releases all locks and cannot obtain a new lock
- Deadlock – When 2 transactions wait indefinitely for each one to release a lock – deadly embrace

Deadlock Control techniques:

- Deadlock prevention – a transaction requiting a lock is aborted when there is a possibility for a deadlock
- Deadlock detection – DBMS periodically tests the database for deadlocks, if found, one transaction is aborted
- Deadlock avoidance – A transaction must obtain all the locks it requires before executing

Database Recovery

- Write-ahead logging – transaction logs are written before execution
- Redundant transaction logs – redundant copies of logs are kept

Wait/Die or Wound/Wait Schemes

Wait/Die

- Older transaction requesting lock waits until newer transactions releases the lock
- Younger transaction requesting lock dies and rescheduled

Wound/Die

- Older transaction requesting lock wounds the newer transaction. Newer transaction is rescheduled
- Newer transaction requesting lock waits until the older transaction releases the lock

Database Recovery using the deferred write/update process:

- Identify the last checkpoint in the DB (when the transactions were last saved to a physical disk)
- Transactions saved before the checkpoint need not be recovered
- For transactions committed after the checkpoint, the DBMS uses transaction logs to redo the transactions in ascending order (oldest to newest)
- Transactions that were rolled back or not committed after the checkpoint need not be restored

Database Recovery using the write through/immediate update process:

- Identify the last checkpoint
- Transactions saved before the checkpoint need not be recovered
- For transactions committed after the checkpoint, the DBMS uses transaction logs to redo the transactions in ascending order (oldest to newest)
- Transactions that had a rollback or not committed after the last checkpoint are rolled back starting with the newest transactions

Chapter 13

- SQL Performance Tuning – client-side
- DBMS Performance Tuning – server-side
- Table space – logical grouping of several data files that store data with similar characteristics
- Data cache/buffer cache – reserved memory space that stores most recently accessed data blocks in RAM
- SQL/Procedure cache – cache for recently executed SQL statements

Create indexes in the following situations:

- Single attributes used in WHERE, HAVING, ORDER BY clauses
- Data sparsity for a column is high
- On join columns other than the PK/FK

Some guidance

- Numeric field comparisons are faster than character, date and NULL comparisons
- Equality comparisons are faster than the inequality comparison
- When using AND conditions, write the condition which is likely to be false first
- When using multiple OR conditions, write the statement which is likely to be false first

Chapter 14 – Distributed Databases

Changes that affected database development:

- Globalisation of business operations
- Customer and market demands favoured on-demand transaction style (web-based)
- Rapid social and technological changes
- Data realms are converging in the digital world

Problems with centralised databases

- Performance degradation as remote locations increase
- High costs of maintaining centralised mainframe systems
- Reliability problems create by SPOF at the central site
- Scalability problems associated with physical limits – space, power, temperature, HVAC
- Organisational rigidity imposed by the database – not agile

Advantages of distributed Databases

- Data is located near the greatest demand site
- Faster data access
- Faster processing
- Growth facilitation
- Improved communications
- Reduced operating costs
- User friendly interface
- Less risk of SPOF
- Processor independence

Disadvantages of distributed Databases

- Complexity of management and control
- Potential reduction of security
- Lack of standards
- Increased storage requirements
- Increased training costs
- Increased costs

- Distributed processing – database's logical processing is shared among 2 or more physically independent sites. Uses only a single site database

- Distributed database – stored a logically related database on 2 or more physically independent sites

- Single site processing, single site data – processing is done on a central system. End users connect using dumb terminals

- Multi-site processing single site data – processing at multiple sites, sharing a single repository

- Multi-site processing, multi-site data – Fully distributed databases with distributed processing

- Homogeneous DDMBMS – Supports only one type (platform) of DBMS to be distributed

- Heterogeneous DDBMS – supports many types (platforms) of DBMSes to be distributed. (With restrictions)

DDBMS Transparency Features

- Distribution transparency
    - Fragmentation transparency – highest form of transparency. User or programmer does not need to know that a database is fragmented
    - Location fragmentation – The user or programmer specifies the fragment names but doesn't need to know where the data is located
    - Local mapping transparency – The end user or programmer must specify both the fragment names and their locations.
- Transaction Transparency – a transaction can update at several sites. Ensures integrity and consistency.
- Failure Transparency – system will continue to operate in the case of a node failure
- Performance Transparency – system will perform if it is centralised
- Heterogeneity Transparency – allows integration of multiple

Data Fragmentation

- Horizontal fragmentation – division of rows into subsets which are fragmented across nodes
- Vertical fragmentation – division of relations into attribute subsets which are fragmented across nodes
- Mixed fragmentation – a combination of horizontal and vertical fragmentation

- Mutual consistency rule – requires that all copies of fragmented data are identical

CAP Theorem

In a highly distributed database there are three desired properties: Consistency, Availability and Partition Tolerance. Although it must be stated that any distributed database can only achieve two of these properties at a time.

- Consistency – plays a big role. All nodes should see the data at the same time which means that replication must be in real-time.
- Availability – A request must always be fulfilled by a database
- Partition Tolerance – The system must continue to operate even following a node failure.

CODD's 12 Commandments

- Local site independence, central site independence, Failure independence
- Location transparency, Fragmentation transparency, replication transparency
- Distributed query processing, hardware independence, operating system independence

- Network independence, Database independence

Cloud Benefits

- Cost-effectiveness
- Scalability
- Latest software
- Mobile Access

- Cloud services support the Availability and Partial Tolerance properties of the CAP theorem

Chapter 15 – Databases for Decision Support

- Business Intelligence – a comprehensive, cohesive and integrated set of tools and processes used to capture, collect, integrate, store and analyse data with the purpose of generating and presenting information to support business decision-making.
- Business intelligence provides a framework for:
    - Collecting and storing operational data
    - Aggregating the operational data into decision support data
    - Analysing the decision support data to generate information
    - Making business decisions
    - Monitoring results to evaluate outcomes of business decisions
    - Predicting future behaviours and outcomes with a high degree of accuracy
- Master Data Management – Collection of concepts, techniques, and processes for proper identification, definition and management of data elements within an organisation
- MDM's main objective – a consistent definition of data throughout the organisation
- Governance – the method or process of government

Business Intelligence Benefits

- Integrating architecture – integrates various solutions across platforms
- Common user interface for data reporting and analysis
- Common data repository fosters a single version of company data
- Improved organisational performance (added efficiency, reduced waste, increase of the bottom line)

Operational and DSS data can be differentiated by timespan, granularity and dimensionality

	Operational Data	**DSS Data**
Timespan	Short	Long
Granularity	Low	High (drill-down/roll-up)
Dimensionality	Single focus	Many dimensions

- Data Warehouses are Integrated, Subject-oriented, Time-variant and Non-volatile

- Data Mart – small, single-subject data warehouse subset that provides decision support to a small group of people. Only difference between data mart and warehouse is the size and scope of the problem being solved.
- Star Schema – data modelling technique used to map multi-dimensional decision support data into a relational database
- Star schema consists of:
    - Facts – Numeric measurements that represent a specific business aspect or activity
    - Dimensions – Qualifying characteristics that provide additional perspectives to a given fact
    - Attributes – Each dimension table contains attributes
    - Attribute Hierarchies – Provides a top-down data organisation used for aggregation and drill down/roll-up
- Study star schema diagram on page 764

- Techniques to enhance Data Warehouse design
    - Normalising dimensional tables
        - Normalising all dimensions into 3NF form. Each dimension table has its own table
        - Snowflake schema – type of star schema wherein each dimension table has its own dimension tables
    - Maintaining multiple fact tables to represent different aggregation levels
        - Aggregate tables are pre-computed at the data-loading phase rather than at run-time
    - Denormalising fact tables
        - Improves speed and reduces storage
    - Partitioning and replicating table
        - Partitioning splits a table into subsets of rows or columns and places the subsets close to the client computer to improve access time
        - Replication makes a copy of a table and places it in a different

Chapter 16

- API – a set of routines, protocols and tools for building software applications

ODBC Components

- ODBC API – Application through which applications access ODBC functionality
- ODBC Driver manager – managed of database connections
- ODBC Driver – connects directly to the DBMS

- DAO, RDO and ODBC do not provide support for non-relational data

OLE-DB Components

- Consumers – objects (applications/processes) that request and use data
- Providers - manage the connection with a data source and provides data to consumers
- Data providers provide data to other processes
- Service providers provide additional functionality to consumers

- OLE-DB does not support scripting

- Active X Data Objects (ADO) supports scripting

Benefits of Internet Technology

- Hardware and software independence
- Common and simple user interface
- Location independence
- Rapid development at manageable costs

- Server-side extension (Wweb-to-Database middleware) – program that interacts directly with the web server to handle specific types of requests

- Study OLE-DB Diagram on page 809

Web servers use a common way to communicate to backend databases:

- CGI
    - Main disadvantage – uses an external script that has to be executed for each request. Degrades performance
- API
    - Implemented as a shared code or DLL. Loaded in memory and available for use on-demand.
    - Disadvantage – shares memory with web server. An API failure might lead to server failure
    - Webserver and operating system-dependent

Cloud Computing

- A computing model for enabling ubiquous, convenient, on-demand network access to a shared pool of configurable computer resources that can rapidly be provisioned and released with minimal management effort
- Cloud computing removes financial and technological barriers so organisations can leverage database technologies in their business processes with minimal cost

Cloud Implementation Types

- **Public Cloud** – Set up by a 3rd party service provider to provide cloud services to the general public. Most common.
- **Private Cloud** – setup by an organisation for its sole use. Geographically dispersed organisations
- **Community Cloud** – built by/for a specific group of organisations that share a common trade

Cloud Services Characteristics

- Ubiquous access via internet technologies
- Shared infrastructure
- Lower costs and variable pricing
- Flexible and scalable services
- Dynamic provisioning
- Service orientation
- Managed operations

Cloud Service

- SaaS – Google docs, Office Live – Turnkey apps that run in the cloud. Clients cannot change actual application. App shared amongst many customers
- PaaS – provider provides capability to build and deploy consumer created applications in the cloud. Customers can build, deploy and manage applications but cannot change underlying infrastructure. Ms Azue
- IaaS – Providers provide capability for customers to provision resources on demand (servers, storage, databases, processing units, virtualised desktops)

Advantages

- Low initial cost
- Scalability
- Ubiquitous computing
- Reliability and performance
- Fast provisioning
- Managed infrastructure

Disadvantages

- Security, privacy, compliance
- Hidden costs of implementation
- Difficult and lengthy data migration
- Complex licensing schemes
- Loss of ownership/control
- Difficult integration with internal systems

- Semantic web – a mechanism for describing concepts in a way that computers can actually understand