Oliver belanger data base systems

Vocab chapter 1

Application program
- Accesses the database by sending queries or requests for data to DBMS

Database
- A collection of related data
- Represents some aspect of the real world universe of discourse (UoD)
- Logically coherent with inherent meaning
- Designed built and populated for a specific purpose

(DBMS) Database management system
- A collection of programs that enable users to create and maintain a database

Database system
- Database and DBMS software combined

(GIS) Geographic information systems
- Stores and accesses maps, weather data and satellite images

Meta-data
- The database definition or descriptive information is also stored by DBMS in the form of a database catalog or dictionary

Multimedia databases
- Information stored and accessed are like pics, sound, vids

(OLAP) Online analytical processing
- Extract and analyze useful business information

Traditional database application
- Information stored and accessed either textual or numeric

Database
- A collection of related data
- Represents some aspect of the real world universe of discourse (UoD)
- Logically coherent with inherent meaning
- Designed built and populated for a specific purpose

(DBMS) Database management system
- A collection of programs that enable users to create and maintain a database
  - Defining a database
    - Involves specifying the data types, structures and constraints of the data to be stored
  - Meta-data
    - The database definition or descriptive information is also stored by DBMS in the form of a database catalog or dictionary
  - Constructing
    - The process of storing data on a storage medium that is controlled by the DBMS
  - Manipulating
    - A database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the UoD and generating reports from data
  - Sharing
    - A database allows multiple users and programs to access the database simultaneously
  - Protection
    - System protection
      - Protects from hardware or software malfunction
    - Security protection
      - Protects against unauthorized or malicious access

Maintenance
Allows system to evolve as requirements change over time

Application program
Accesses the database by sending queries or requests for data to DBMS
Query
Causes some data to be retrieved
Transaction
May cause some data to be read and some data to be written into the database

Database system
Database and DBMS software combined

virtual data
Data that is derived from database file but no explicitly stored
Meta-data
Describes structure of the primary database
Data abstraction
Program-data independence
Program-operation independence

SQL
4.3
Basic retrieval queries
Multiset
Select statement
Three components +1
Select <attribute list>
Attributes to be selected by SQL
*
Asterisks <*> selects all attributes
.
Qualifier
Table1.name
Select name attribute in table1
ALL
Explicitly selects duplicate as well as unique values, unneeded
Distinct
Selects only unique values in attributes selected
From <table list>
Table to take those attributes from

Where <condition>
Boolean expression that identifies the tuples to be retrieved
=
<attribute> = ‘value’
Will select all tuples that equal this value in this attribute
<attribute> = <attribute>
Joint condition
Combines two tuples in different tables that are joint
AND
Used as a continuation of the boolean expression
<name> = ‘jay’ AND <place> = ‘paris’

Thus the selected tuple will be one where name is jay and place is paris

AS

Used for aliasing

<attribute A>AS A

Can now use A as a qualifier to select attributes

A.name is the same as attributeA.name

Blanc

If there is no where clause then the resulting table Is the cross product, all possible combination of selected attributes and tables

LIKE

Used when we want to pair strings, so like everyone with this exact string in this attribute denoted by like and %

WHERE: Address LIKE '%Houston,TX%';

WHERE: <attribute> LIKE '%<string>%';

Can be used with single characters as well where unused characters are denoted by _ for all people born in 1950

WHERE: Address LIKE '<_ _ 5 _ _ _>';

If '' are necessary in the pairing they must be doubled

If % or _ is necessary then an escape character is used

In 'AB_CD\%EF' ESCAPE '\' means 'AB_CD%EF'

Between

Used in correspondence with and, will select the tuples that fall between the condition

WHERE: (Salary BETWEEN 3000 AND 4000) and Dno=5;

WHERE: <attribute> BETWEEN <value> AND <value>;

ORDER BY

The order in which the values are set alpha numerically

In this case first by department numerically then by alphabetical order last and first name

ORDER BY D.Dname, E.Lname, E.Fname;

ORDER BY <attribute>, <attribute>,<attribute>;

The key word DESC can used to order in descending order and ASC in ascending though ASC is default

Arithmetic can be used with attribute with numerical values

We can use +,-,*,/ in the same way we would normally

ll can be used to append two strings values

+ and - can be used to increment and decrement date, time or timestamp by an interval

4.4

Insert delete and update statements in sql

INSERT

Form 1

Simples form used to add a tuple to a relation

Must be set in the same order of attributes as the original table

U1:	INSERT INTO	EMPLOYEE
	VALUES	('Richard','k','marini','653298653','1962-12-30','98 Oak Forest, Katy, TX', 'm', 37000, '653298653', 4);
U1:	INSERT INTO	<table>
	VALUES	('<value>','<value>','<value>'.....'<value>');

Form 2

Add tuple into a relation

Attributes that are allowed null or have a default value can be left out

```
U1:  INSERT INTO   EMPLOYEE(Fname, Lname, Dno, Ssn)
     VALUES        (‘Richard’,’k’,‘marini’, 4, ‘653298653’);

U1:  INSERT INTO   EMPLOYEE(<attribute>,<attribute> , <attribute>)
     VALUES        (‘<value>’,’<value>’, ‘<value>’);
```

Form 3

Insert multiple tuple in conjunction with creating a new relation

```
U3A: CREATE TABLE     WORKS_ON_INFO
     (  Emp_name      VARCHAR(15),
        Proj_name     VARCHAR(15),
        Hours_per_week DECIMAL(3,1));
U3B: INSERT INTO      WORKS_ON_INFO (Emp_name, Proj_name,
                      Hours_per_week)
     SELECT           E.Lname, P.Pname, W.Hours
     FROM             PROJECT P, WORKS_ON W, EMPLOYEE E
     WHERE            P.Pnumber= W.Pno AND W.Essn=E.Ssn;
```

Delete

Deletes tuple from relation

```
U4A: DELETE FROM   EMPLOYEE
     WHERE         Lname=‘Brown’;
```

Deletes all tuples with last name brown

```
U4A: DELETE FROM   EMPLOYEE
     WHERE         Ssn=‘123456789’;
```

Deletes all tuples with ssn ^

```
U4A: DELETE FROM   EMPLOYEE
     WHERE         Dno=‘5’;
```

Deletes all tuples with dno 5

```
U4A: DELETE FROM   EMPLOYEE
```

Deletes all tuples in employee but does not delete the table

Update

Updates table

CAN ONLY UPDATE ONE TABLE AT A TIME

```
U5:  UPDATE   PROJECT
     SET      Plocation= ‘Bellaire’, Dnum= 5
     WHERE    Pnumber=10;
U6:  UPDATE   EMPLOYEE
     SET      SALARY = SALARY*1.1
     WHERE    DNO=5
```

5.3

Views (virtual tables) in sql

- A view is a single table that is derived from other tables
  - Not a real table it is a virtual table
    - Not actually stored physically
    - Base tables used called defining tables
  - Command
    - Create view
      - Once created can be interacted with like any other table
      - Views should always be up to date if we modify a base table the view must automatically reflect these changes

```
V1:   CREATE VIEW    WORKS_ON1
      AS SELECT      Fname, Lname, Pname Hours
      FROM           EMPLOYEE, PROJECT, WORKS_ON
      WHERE          Ssn=Essn AND Pno=Pnumber;
V2:   CREATE VIEW    DEPT_INFO(Dept_name,No_of_emps, Total_sal)
      AS SELECT      Dname, COUNT (*), SUM (Salary)
      FROM           DEPARTMENT, EMPLOYEE
      WHERE          Dnumber = Dno
      GROUP BY       Dname;
```

    - Drop view
      - Gets rid of view

```
V1A:  DROP VIEW      WORKS_ON1
```

View implementation, view update and inline views

- Query modification
  - Modifying or transforming the view query into a query on the underlying base table
- View materialization
  - Physically create a temporary view table when the view is first created and keep it on assumption that other queries will follow
    - Strategy to update view when base table changed necessary
      - Incremental update
    - View is kept as long as queried and then it is removed and started from scratch when called again
- View updates can only be made if they only execute only one possible update on the base table
- Summary
  - View with a single defining table is updatable if the view attributes contain the primary key of the base relation as well as all attributes with the not null constraint that do not have default values specified
  - Views defined on multiple tables using joins are generally not updatable
  - Views defined using grouping and aggregate functions are not updatable
- With check option must be added at end of the view definition if a view is to be updated

7.3

Er model

Entities and attributes

- Entity
  - Basic object that ER model represents
  - Thing in real world with an independent existence

- Weak entity types
  - No key attributes

Mapping 9

Mapping of binary 1:1 relationship types

- 3 approaches
  - Foreign key approach
    - Choose one entity with total participation in the relation and include the primary key of the entity that is not the one you chose as a foreign key, include all simple attributes of the 1:1 relation as attributes of the entity you chose
  - Merged relation approach
    - Only possible when both participation is total (both entities have same number of tuples)
      - Merge the two entity types and the relationship into a single relation
  - Cross-reference or relationship relation approach
    - Set up a third relation for the purpose of cross-referencing the primary keys of both entity types
      - This relation includes the primary key attributes of both entities as foreign keys to themselves
      - The primary key of this relation will be one of the two foreign keys and the other will be a unique key of the relation

Mapping of binary 1:N relationship

- 2 approaches
  - First approach
    - Choose the entity that represents the N-side of the relation
    - include the primary key of the other entity on the 1 side as a foreign key
    - Include any simple attribute or simple components of composite attribute of the relationship in the relation
  - Relationship relation (cross reference)
    - Create a separate relation whose attributes are the primary keys of both entities as foreign keys
    - The primary key will be the same as the one from the Nside entity

Mapping of binary M:N relationship

- Relationship relation (cross reference) only option
  - Create new relation
  - Include as foreign key the primary keys of the relations that represent the participating entities
  - Their combination represent the primary key of the relation
  - Include any simple attributes of the relationship as attributes of the relation

- The role of information systems in organizations
  - The organizational context for using database systems
    - Reasons that IT (information technology) and IRM (information resource management) have be recognized as being key to successful business management
      - Data is regarded as a corporate resource
        - Its management and control is considered central to the effective working of the organization
      - More functions in organizations are computerized
        - Increased need to keep large volumes of data available in an up to the minute current state
      - As the complexity of the data and applications grow
        - Complex relationships among data need to be maintained and modeled
      - There is a tendency toward consolidation of information resources in many organizations
      - Many organizations are reducing their personnel costs by letting end users perform business transactions
    - Capabilities provided by database systems that are integral to computer based information systems
      - Integrating data across multiple applications into a single database
      - Support for developing new applications in a short amount of time by using high level languages like sql
      - Providing support for casual access for browsing and querying by managers while supporting major production level transaction processing for customers
    - Use of distributed systems over centralized systems
      - Personal computers and database system like software products are being heavily utilized by users who use to belong to the category of casual and occasional database users
        - Personal databases are becoming popular
      - Distributed and client-server DBMS are making distributing databases over multiple computer systems faster and more secure
        - Users can still access remote data using the facilities provided by the DBMS as a client through the web
      - Organizations use data dictionary systems (information repositories)
        - Mini DBMS that manage meta data
        - Stores and manages
          - Descriptions of the schemas of the database systems
          - Detailed information on physical database design such as storage structures, access paths and file and record sizes
          - Descriptions of the types of database users their responsibilities and their access rights
          - High-level descriptions of the database transactions and applications and of the relationships of the users to transactions
          - The relationship between database transactions and the data items referenced by them. This is useful in determining which transactions are affected when certain data definitions are changed
          - Usage statistics such as frequencies of queries and transactions and access counts to different portions of the database
          - The history of any change made to the database and applications and documentation that described the reasons for these changes. This is sometimes referred to as data provenance

10.1.2
The information system life cycle
Information life cycle is
- Macro life cycle

Database system life cycle Is
- Micro life cycle

Macro lifecycle in order
- Feasibility analysis --- what is feasible and what isnt
  - Concerned with analyzing potential applications areas identifying the economics of information gathering and dissemination performing preliminary cost benefit studies determining the complexity of data and processes and setting up priorities among applications
- Requirements collection and analysis ----- what you need and what you dont
  - Detailed requirements are collected by interacting with potential users and user groups to identify their particular problems and needs. Interapplication dependencies communication and reporting procedures are identified
- Design ----- write it on paper what your gonna do
  - First aspect
    - The design of the database system
  - Second aspect
    - The design of the application systems that use and process the database through retrievals and updates
- Implementation ------ put it together in RL
  - The information system is implemented the database is loaded and the database transactions are implemented and tested
- Validation and acceptance testing ----- does it work
  - The acceptability of the system in meeting user requirements and performance criteria is validated the system is tested against performance criteria and behavior specifications
- Deployment operation maintenance
  - Sending it on the market or place and maintaining and tweaking

Micro life cycle
- System definition
  - Scope of DBS its users and its applications are defined
  - Interface for various categories of users, response time constraints and storage and processing needs are identified
- Database design
  - Complete logical and physical design of the database system on the chosen DBMS is prepared
- Database implementation
  - Process of specifying the conceptual external and internal database definitions creating empty database files and implementing the software applications
- Loading or data conversion
  - Database is populated by either loading the data directly of by converting existing files into the database system format
- Application conversion
  - Old software is converted to new system
- Testing and validation
  - New system is tested and validated
- Operation
  - Put into use
- Monitoring and maintenance
  - Its in the name

10.2
The database design and implementation process

- Phase 1: requirements collection and analysis
  - Major users identified
  - Existing documentation concerning application is studied
  - Current operating environment and planned use of information is studied
- Phase 2: conceptual design
  - Phase 2a: conceptual schema design
    - Goal is a complete understanding of the database structure meaning interrelationships and constraints
    - Stable description of the database contents
      - Must have these elements
        - Expressiveness
          - Distinguish different types of data relationships and constraints
        - Simplicity and understanding
          - Simple enough for non specialist to understand
        - Minimality
          - Few non overlapping basic concepts
        - Diagrammatic representation
          - Diagram thats easy to understand
        - Formality
          - Must represent a formal Unambiguous specification of data
  - Phase 2b: transaction design
    - 80-20 rule
      - 80 percent of the work load is represented by 20 percent of the most frequently used transactions
    - Three category
      - Retrieval transactions
        - Retrieve data for display
      - Update transactions
        - Used to enter in new data or modify existing data
      - Mixed transactions
        - Complex application that do some retrieval and some update
- Phase 3: choice of a DBMS
  - Factors of choosing a DBMS
    - Cost
      - Software acquisition cost
        - Up front cost of buying software
      - Maintenance cost
        - Recurring cost of receiving standard maintenance and service
      - Hardware acquisition cost
        - Up front cost of hardware
      - Database creation and conversion cost
        - Cost of creating database from scratch or converting it from an old one
      - Personnel cost
        - Cost of hiring people to run your new database system
      - Training cost
        - Cost of training people to use and program DBMS
      - Operating cost
        - Cost of operating the DBMS

- Benefits
  - Data complexity
    - As data relationships become more complex the need for a DBMS increases
  - Sharing among applications
    - The need for a DBMS is greater when common data needs to be shared across applications
  - Dynamically evolving or growing data
    - Data that changes constantly is easier to cope with with DBMS
  - Frequency of ad hoc requests for data
    - File system are not all suitable for ad hoc retrieval of data
  - Data volume and need for control
    - The sheer volume of data and the need to control it may require DBMS
- Organizational factors
  - Organization wide adoption of a certain philosophy
    - How does the organization believe data should be represented
  - Familiarity of personnel with the system
    - How familiar the personnel is with any particular DBMS
  - Availability of vendor services
    - How available is vendor assistance in solving problems and such

Phase 4: data model mapping

- System independent mapping
  - A mapping that does not take into account any particular DBMS
- Tailoring the schema to a specific DBMS
  - Self explanatory

Phase 5: physical design

- Response time
  - Elapsed time between submitting a database transaction and receiving a response
- Space utilization
  - Amount of storage space used by the database file and their access path structures
- Transaction throughput
  - Average number of transactions that can be processed per minute

Physical design

Storage medium

- Primary storage
  - Storage media that can be operated on by the CPU
  - Volatile
    - Main memory
    - Cashe memory
- Secondary storage
  - Non removable
  - Non volatile
    - Hard drives
- Tertiary storage
  - Easily removable storage,
  - Non volatile
    - cds and tapes

Disk devices

- Random access
- Each circle is a track
- Each track on different disk is a cylinder
- Each track is broken up into ark sectors
- Track is devised into equal sized disk blocks during disk formatting

- Blocks seperated by interblock gaps which determine which block on the track follows each interblock
- Hardware address of a block combo of cylinder number track number and block number supplied to disk i/o hardware

Magnetic tape storage device
- To access memory we must scan all preceding blocks
- SEQUENTIAL ORDER

Direct organization
- Computer keeps track of storage location of each record using keys, so that data can be retrieved when needed
- New transaction data does not need to be sorted
- Processing that requires immediate responses or updating is easily performed

Sequential organization
- Records are physically stored in a specific order according to a key field in each record
  - Fast and efficient with large volume of data that need to be processed periodically (batch)
  - All new transactions be stored in correct sequential order
  - All interaction require rearranging of files
  - To slow to hand immediate applications

Index sequential
- Physically stored in sequence on a direct access storage device based on key field on each record
- Each file contains an index that references one or more key fields of each data record to its storage location

1st Normal form
- Relational property
  - No duplicate records
  - One primary key
  - Every cell only Single valued
  - Unique name
  - Unique attributes
  - Data must be part of data pool allowed

2nd normal form
- Functional property
  - Primary key must determine non primary key attributes

3rd normal form
- Transitive dependence
  - Each non primary attribute must provide a fact about the key the whole key and nothing but the key