Week 2 - SQL and Conceptual Data Design

- SQL Introduction
  - DDL
    - Data Definition Language
    - Create, drop or alter the relation schema
  - DML
    - Data Manipulation Language
    - The retrieval of information stored in the database
    - A Query is a statement requesting the retrieval of information
    - The portion of a DML that involves information retrieval is called a query language
    - The **insertion** of new information
    - The **deletion** of information
    - The **modification** of information
  - DCL
    - Data Control Language
    - Commands that control a database, including administering privileges and users
  -
- SELECT
  - Clauses of the SELECT statement:

▸ SELECT	Lists the columns (and expressions) that should be returned from the query
▸ FROM	Indicate the table(s) from which data will be obtained
▸ WHERE	Indicate the conditions to include a tuple in the result
▸ GROUP BY	Indicate the categorization of tuples
▸ HAVING	Indicate the conditions to include a category
▸ ORDER BY	Sorts the result according to specified criteria

  - a Select-From-Where (SFW) query is equivalent to the relational algebra expression: $\Pi_{A1, A2, ..., An} (\sigma_{condition} (R_1 \times R_2 \times ... \times R_m))$
  - ■ List the names of all Australian students.

    ```
    SELECT name FROM Student WHERE country='AUS'
    ```

  - ■ Corresponding relational algebra expression

    $\pi_{name} (\sigma_{country='AUS'} (Student))$

  - SQL does not permit the '-' character in names, and they are case insensitive, thus you can use capitol or small letters
- ORDER BY
  - There are two options
    - Two options (per attribute):
      - ▸ ASC ascending order (default)
      - ▸ DESC descending order
- Duplicates
  - SQL allows duplicates in relations as well as in query results
  - To force elimination of duplicates, insert the keyword **distinct** after **select**
  - To force all duplicates NOT to be removed, use the keyword **all** after select
- Arithmetic Expressions
  - Use "SELECT *" to selected all attributes
- The Rename Operation
  - SQL allows renaming relations and attributes using the ss clause:
    - old_name **as** new_name
- WHERE
  - Specifies conditions that the result must satisfy

Comparison operators in SQL: = , > , >= , < , <= , != , <>

- Comparison results can be combined using the logical connectives **and**, **or**, and **not**.

```
SELECT sid
  FROM Assessment
 WHERE uos_code = 'COMP5138'   AND
       mark BETWEEN 75 AND 100
```

- String Operations
  - SQL includes a string-matching operator for comparisons on character strings
  - Patterns are described using two special characters ("wildcards"):
    - percent (%). The % character matches any substring.
    - underscore (_). The _ character matches any character.
  - SQL supports a variety of string operations such as
    - concatenation (using "||")
    - converting from upper to lower case (and vice versa)
    - finding string length, extracting substrings, etc.
- Regular Expressions Matches
  - What are regular expressions?
    - Pattern consisting of *character literals* and/or *metacharacters*
    - *metacharacters* specify how to process a regular expression
      - ( ) grouping
      - | alternative
      - [ ] character list
      - . matches any character
      - * repeat preceeding pattern zero, one, or more times
      - + repeat preceeding pattern one or more times
      - ^ start of a line
      - $ end of line
  - Example:

```
select title
  from UnitOfStudy
 where regexp_like(uos_code,'^COMP[:digit:]{4}')
```

- Date and Time in SQL

SQL Type	Example	Accuracy	Description
DATE	'2012-03-26'	1 day	a date (some systems incl. time)
TIME	'16:12:05'	ca. 1 ms	a time, often down to nanoseconds
TIMESTAMP	'2012-03-26 16:12:05'	ca. 1 sec	Time at a certain date: SQL Server: DATETIME
INTERVAL	'5 DAY'	years - ms	a time duration

Main Operations

- **EXTRACT(** *component* **FROM** *date* **)**
  - e.g. EXTRACT(year FROM enrolmentDate)
- **DATE** *string* (Oracle syntax: TO_DATE(*string,template*))
  - e.g. DATE '2012-03-01'
  - Some systems allow templates on how to interpret *string*
  - Oracle syntax: TO_DATE('01-03-2012', 'DD-Mon-YYYY')
- **+/- INTERVAL**:
  - e.g. '2012-04-01' + INTERVAL '36 HOUR'

- FROM
  - The **from** clause lists the relations involved in the query
  - Corresponds to the Cartesian product operation of the relational algebra

- Join-predicates must be explicitly stated in the where clause

- Joins and Aggregate Function
  - Joins
    - Join
      - A relational operation that causes two or more tables with a common domain to be combined inrto a single table or view
    - Equi-join
      - A join in which the joining condition is based on equality between values in the common columns; common columns appear redundantly in the result table
    - Natural join
      - An equi-join in which one of the duplicate columns is eliminated in the result table
    - Outer join
      - A join in which rows that do not have matching values in common columns are nonetheless included in the result table
    - Union join
      - Includes all columns from each table in the join, and an instance for each row of each table

SQL offers join operators to directly formulate the natural join, equi-join, and the theta join RA operations.

- R **natural join** S
- R **inner join** S **on** <join condition>
- R **inner join** S **using** (<list of attributes>)

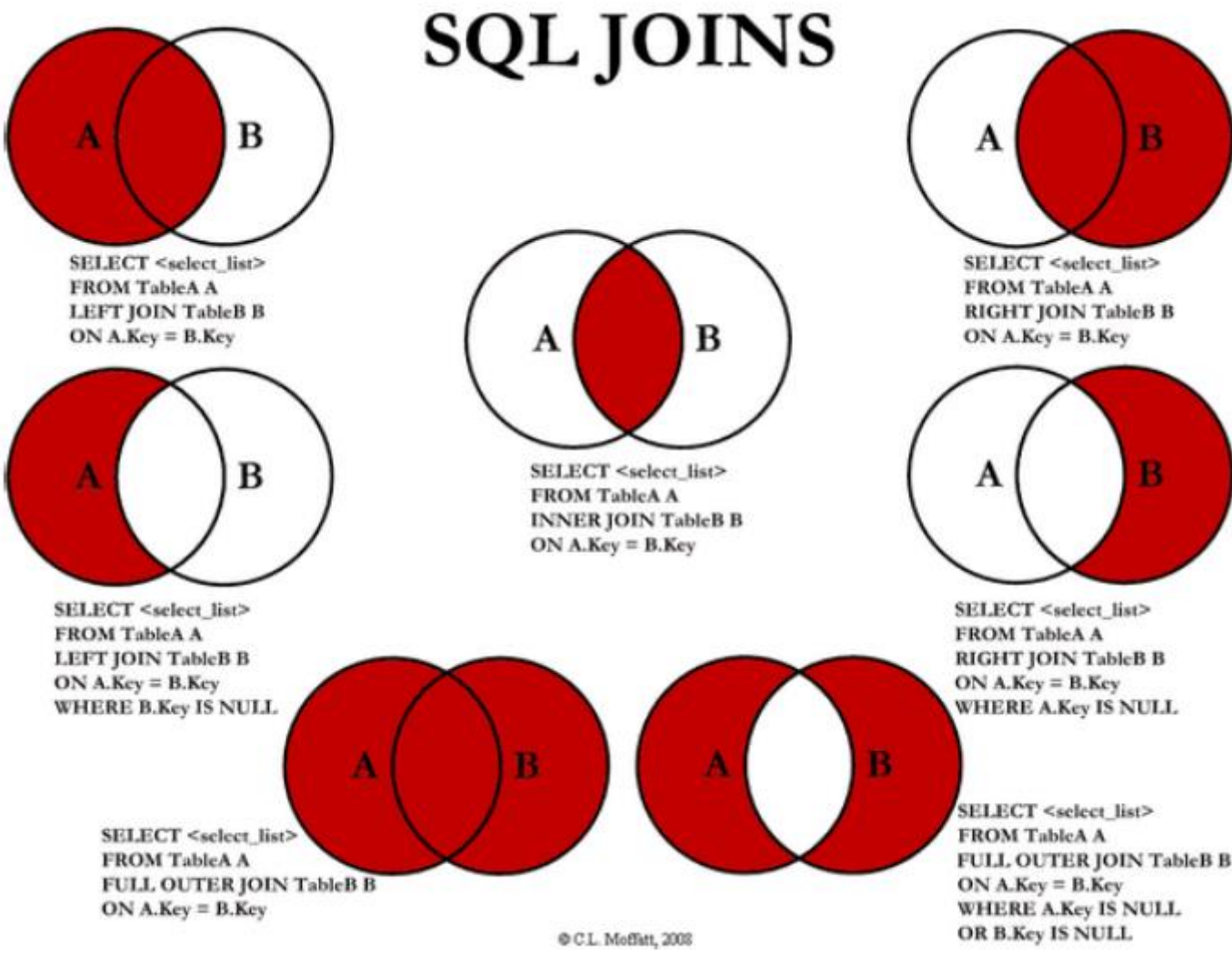
  - These additional operations are typically used as subquery expressions in the from clause

- List all students and in which courses they enrolled.

```
select name, uos_code, semester
  from Student natural join Enrolled
```

- Who is teaching "ISYS2120"?

```
select name
  from UnitOfStudy inner join Academic on lecturer=empid
 where uos_code='ISYS2120'
```



- Aggregate Functions
  - These functions operate on the multiset of values of a column of a relation, and return a value
  - These functions operate on the *multiset* of values of a column of a relation, and return a value
    - **avg:** average value
    - **min:** minimum value
    - **max:** maximum value
    - **sum:** sum of values
    - **count:** number of values
    - Must use **distinct** in addition to aggregate over *sets*
- NULL Values
  - It is possible for tuples to have a null value, denoted by **null**, for some attributes
  - The predicate '**is null**' can be used to check for null values

- Conceptual Database Design using the Entity Relationship Model
  - Database design sequence
    - Requirements Analysis
      - Understand what data is stored, what applications must be built, what operations are most frequent
    - Conceptual Design
      - Develop high-level description of the data closely matching how users think of the data
    - Logical Design
      - Convert conceptual design into a logical database schema
    - Physical Design
      - Logical schema into a physical schema for a specific DBMS and tuned for app
  - Conceptual Data Model
    - Goal
      - Specification of database schema

- Methodology
  - **Conceptual Design**: A technique for understanding and capturing business information requirements graphically
- CDD does not imply how data is implemented, created, modified, used or deleted
- A CDD is model & database independent
- Entity-relationship model (ERM)
- Object oriented Data Models

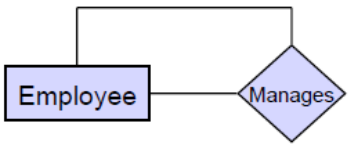
○ Entity Relationship Model

Appendix: Notation Comparison

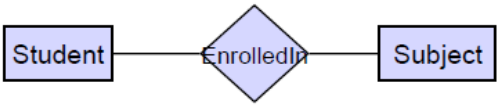
	Kifer / Bernstein / Lewis (ISYS2120)	Ramakrishnan / Gehrke	Ullman / Widom	Korth/ Silberschatz / Sudarshan	Hoffer / Prescott "Crows-Foot"
Entity Name	Entity Type	Entity Set *(plural names)*	Entity Set *(plural names)*	Entity Set	Entity Type
Attributes	attr attr E *only atomic; single-|set-valued*	attr E *only atomic & single valued*	attr E *only single valued (but mention variants with structs & sets)*	attr attr E *single-|set-valued composite attr. derived attributes*	E attr attr *single-|set-valued composite attr. derived attributes*
Key Constraints (1-many relationship)	*(arrow from N-side to diamond)*	*(arrow from N-side to diamond)*	*(arrow from diamond to 1-side)*	*(arrow from diamond to 1-side)*	E1 E2 *(no diamond;tick on 1-side, crow's foot on many side)*
Participation Constraints	*(thick line on total participation side)*	*(thick line on total participation side)*	*n/a*	*(double line - total participation side)*	E1 E2 or E1 E2
Cardinality Constraints	1..* 0..5 *min..max notation*	*n/a*	≥1 ≤5 *limit constraint*	1..* 0..5 *min..max notation*	E1 0..5 E2 *on opposite side!*
Roles	yes	yes	yes	yes	yes
Weak Entity (& identifying rel.ship)					
ISA	ISA	ISA	ISA	ISA	

- Entity
  - A person, place, object, event or concept about which you want to gather and store data
- Entity Type
  - A collection of entities that share common properties or characteristics
- Attribute
  - Describes one aspect of an entity type
- Domain
  - Possible values of an attribute
- Key
  - Minimal set of attributes that uniquely identifies an entity in a set
    - E.g primary key, foreign key
- Relationship
  - Relates two or more entities
- Relationship Type
  - Set of similar relationships

▶ Unary Relationship (Recursive)



▶ Binary Relationship (most common kind of relationship)



▶ Ternary Relationship
(three entity types involved)