

ITM 500: Data & Information Management

TOPICS TO REVIEW:

- 1- INTRODUCTION OF DBMS & DEFINITIONS USED IN DBMS
- 2- SQL COMMANDS, ORDERS OF COMMANDS AND CONDITIONS

Reference:

- (1) Definitions, examples and demonstrations:
Database Systems: Design, Implementation, and Management, 13th Edition.
- (2) Demonstrations and examples:
<https://www.w3schools.com/sql/default.asp>

ITM 500: Data & Information Management

INTRODUCTION OF DBMS & DEFINITIONS USED IN DBMS

DBMS (Database Management System)

- Software applications for database management (Organizing and Manipulate data)
- Ex: My SQL

Servers:

- Accommodate users to varies DB

Types of Database:

- 1- Single-user DB
Ex: DB on PC (1 DB, 1 User)
- 2- Multi-user DB
Ex : FB, Youtube, IG,... (1 DB, \geq 1 User)
- 3- Classification by location
 - a- Centralized DB (Data kept in 1 site)
 - b- Distributed DB (Data kept across different sites)
 - c- Cloud DB (Sam as b- but not just for 1 organization)

Classification by type:

- 1- General purpose DB
- 2- Discipline specific
- 3- Operation: Never shut down

Analytical DB

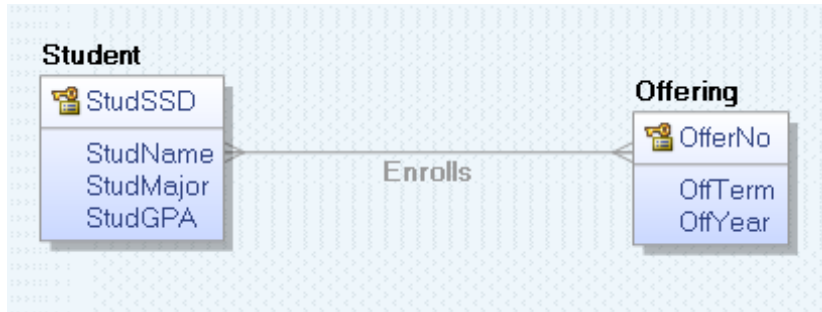
- Data warehouse
- OLAP (Online analytical processing)
- BI (Business intelligence)

Degree of which data is structured:

- Unstructured
- Structured
- Semi Structured

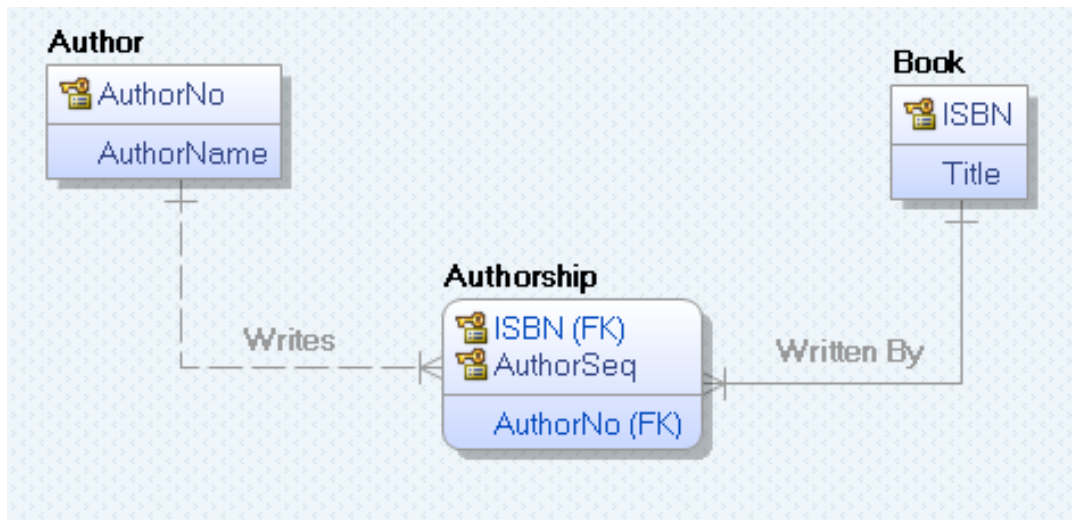
CHAPTER 2: DATA MODELS

ITM 500: Data & Information Management



- Entities: Student, Offering
- Attributes of each Entities: Student (StudSSD, StudName, StudMajor, StudGPA)
- Primary key of Entity "Student": StudSSD
- Type of relationship between these Entities: M:M
- Other types:
 - 1- 1:1 or 1...1
 - 2- 1:M or 1...*
 - 3- M:M or *...*

More examples:



The Author Entity contains

2 attributes AuthorNo, AuthorName

AuthorNo is the primary key for the Author

ITM 500: Data & Information Management

The Book Entity contains

2 attributes ISBN, Title

ISBN is the primary key of the Book

The Authorship Entity contains

a. 3 attributes – ISBN, AuthorSeq, and AuthorNo

b. ISBN + AuthorSeq is the alternate composite primary key of Authorship

A Book may have 1 or more Authorships

An Author may have 1 or more Authorships

ITM 500: Data & Information Management

SQL COMMANDS, ORDERS OF COMMANDS AND CONDITIONS – with examples

WEEK 3 & 4: CHAPTER 7

- Data types and functions
- Simple SQL formation
- Aggregates, Joins and DML (Data Manipulation Language) commands

A- BASIC SQL SYNTAX FORMATION

SYNTAX	EX
SELECT [attributes] FROM [one or more table(s)] WHERE [condition(s)];	SELECT Stud_name FROM Stud WHERE Stud_name LIKE '%Anh%';

Stud
Stud_name
Stud_ID
Stud_courses
Grad_Yr

Depend on which applications, ' ; ' might or might not required at the end of your code.

B- DATA TYPES IN SQL

CHARACTER [(length)] or CHAR [(length)]	Character strings, including Unicode, of a fixed length . The default length is 1. Ex: CHAR(10) or CHARACTER(10) Valid notation: 'Race car', '24865', '1998-10-25' (Notice the '')
VARCHAR (length)	Character strings, including Unicode, of a variable length is up to the maximum length specified in the data type declaration
BOOLEAN	Including two values: TRUE or FALSE. Valid notation: TRUE true True False

ITM 500: Data & Information Management

SMALLINT	<p>Numeric values with an implied scale of zero. It stores any integer value between the range 2^{-15} and $2^{15} - 1$.</p> <p>Valid notation:</p> <p>-32768 0 -30.3 (digits to the right of the decimal point are truncated) 32767</p>
INTEGER or INT	<p>Numeric values with an implied scale of zero. It stores any integer value between the range 2^{-31} and $2^{31} - 1$.</p> <p>Valid notation:</p> <p>-2147483648 -1025 0 1025.98 (digits to the right of the decimal point are truncated) 2147483647</p>
DECIMAL [(p[,s])] or DEC [(p[,s])]	<p>Numeric values, for which you may define a precision and a scale in the data type declaration</p> <p>Can be declared in one of three different ways:</p> <ul style="list-style-type: none"> DECIMAL - Precision defaults to 38, Scale defaults to 0 DECIMAL(p) - Scale defaults to 0 DECIMAL(p, s) - Precision and Scale are defined by the user <p>Valid notation:</p> <p>1234567 1234567.123 1234567.1234 (Final digit is truncated) -1234567 -1234567.123 -1234567.1234 (Final digit is truncated)</p>
NUMERIC [(p[,s])]	Data type is the same as the DECIMAL data type
REAL	<p>To represent approximate numeric values</p> <p>Valid notation:</p> <p>-2345 0 1E-3 1.245 123456789012345678901234567890</p>
FLOAT(p)	<p>Valid notation:</p> <p>12345678</p>

ITM 500: Data & Information Management

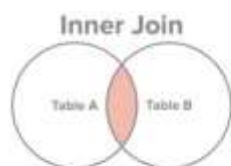
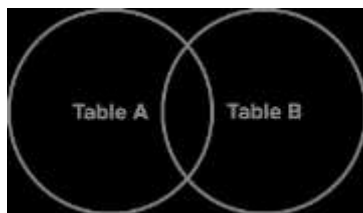
	1.2 123.45678 -12345678 -1.2 -123.45678
DATE	Valid notation: DATE '1999-01-01' DATE '2000-2-2' date '0-1-1'
TIME	Valid notation: TIME '00:00:00' TIME '1:00:00' TIME '23:59:59' time '23:59:59.99'
TIMESTAMP	Valid notation: TIMESTAMP `1999-12-31 23:59:59.99` TIMESTAMP `0-01-01 00:00:00`

ITM 500: Data & Information Management

<i>Data Type Families and Data Types</i>	
Data Type Family	Data Types
Character String	CHARACTER, VARCHAR, CLOB
Boolean	BOOLEAN
Binary String	BLOB
Date Time	DATE, TIME, TIMESTAMP
Number	SMALLINT, INTEGER, DECIMAL, NUMERIC, REAL, FLOAT, DOUBLE

C- AGGREGATES AND JOIN FUCTIONS:

a- Types of JOIN:



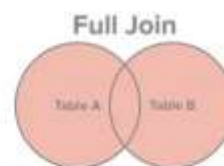
Select all records from Table A and Table B, where the join condition is met.



Select all records from Table A, along with records from Table B for which the join condition is met (if at all).




Select all records from Table B, along with records from Table A for which the join condition is met (if at all).



Select all records from Table A and Table B, regardless of whether the join condition is met or not.

** Recursive JOIN  A query that joins a table to itself

** Cross JOIN  A query that select all attributes user want from no united tables

ITM 500: Data & Information Management

Sample table:

first_name	last_name	order_date	order_amount
George	Washington	07/04/1776	\$234.56
Thomas	Jefferson	03/14/1760	\$78.50
John	Adams	05/23/1784	\$124.00
Thomas	Jefferson	09/03/1790	\$65.50
NULL	NULL	07/21/1795	\$25.50
NULL	NULL	11/27/1787	\$14.40
James	Madison	NULL	NULL
James	Monroe	NULL	NULL

(Inner) join:

```
select first_name, last_name, order_date, order_amount
from customers c
inner join orders o
on c.customer_id = o.customer_id
```

Left (outer) join:

```
select first_name, last_name, order_date, order_amount
from customers c
left join orders o
on c.customer_id = o.customer_id
```

Right (outer) join:

```
select first_name, last_name, order_date, order_amount
from customers c
right join orders o
on c.customer_id = o.customer_id
```

Full (outer) join:

```
select first_name, last_name, order_date, order_amount
from customers c
full join orders o
on c.customer_id = o.customer_id
```

ITM 500: Data & Information Management

b- Some common aggregates commands:

1- ORDER BY: - Example

<code>SELECT * FROM Customers ORDER BY CustomerName;</code>	sorted by the "CustomerName" column
<code>SELECT * FROM Customers ORDER BY CustomerName ASC;</code>	sorted ascending by the "CustomerName" column
<code>SELECT * FROM Customers ORDER BY CustomerName DESC;</code>	sorted descending by the "CustomerName" column

2- WHERE (To add conditional restriction)

`SELECT [columnlist]`

`FROM [tablelist]`

`WHERE [conditionlist]`

3- COMPARISON OPERATIONS

Operator	Description
=	Equal to.
>	Greater than.
<	Less than.
>=	Greater than equal to.
<=	Less than equal to.
<>	Not equal to.

ITM 500: Data & Information Management

4- LOGICAL OPERATORS: AND, OR, NOT

<pre>SELECT column1, column2, ... FROM table_name WHERE condition1 AND condition2 AND condition3 ...;</pre>	displays a record if all the conditions separated by AND are TRUE
<pre>SELECT column1, column2, ... FROM table_name WHERE condition1 OR condition2 OR condition3 ...;</pre>	Displays a record if any of the conditions separated by OR is TRUE.
<pre>SELECT column1, column2, ... FROM table_name WHERE NOT condition;</pre>	displays a record if the condition(s) is NOT TRUE.

5- Special Operators: BETWEEN, IN, LIKE, IS NULL

LIKE	column value is similar to specified character(s).
IN	column value is equal to any one of a specified set of values.
BETWEEN...AND	column value is between two values, including the end values specified in the range.
IS NULL	column value does not exist.

6- Basic SQL Aggregate functions: COUNT, MIN, MAX, SUM, AVG

<pre>SELECT COUNT(column_name) FROM table_name WHERE condition;</pre>	returns the number of rows that matches a specified criteria.
<pre>SELECT AVG(column_name) FROM table_name WHERE condition;</pre>	returns the average value of a numeric column
<pre>SELECT SUM(column_name) FROM table_name WHERE condition;</pre>	returns the total sum of a numeric column
<pre>SELECT MIN(column_name) FROM table_name WHERE condition;</pre>	returns the smallest value of the selected column
<pre>SELECT MAX(column_name) FROM table_name WHERE condition;</pre>	returns the largest value of the selected column

ITM 500: Data & Information Management

7- GROUP BY statement

```
SELECT column_name(s)
FROM table_name
WHERE condition

GROUP BY column_name(s)

ORDER BY column_name(s);
```

Sample table:

CustID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

Ex1

```
SELECT COUNT(CustomerID), Country
```

(1) Database Systems: Design, Implementation, and Management, 13th Edition.

ITM 500: Data & Information Management

```
FROM Customers  
GROUP BY Country;
```

Ex2:

```
SELECT COUNT(CustomerID), Country  
FROM Customers  
GROUP BY Country  
ORDER BY COUNT(CustomerID) DESC;
```

1- HAVING clause

HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

```
SELECT column_name(s)  
FROM table_name  
WHERE condition  
  
GROUP BY column_name(s)  
  
HAVING condition  
  
ORDER BY column_name(s);
```

Sample table:

OrderID	CustomerID	EmployeeID	OrderDate	ShipperID
10248	90	5	1996-07-04	3
10249	81	6	1996-07-05	1
10250	34	4	1996-07-08	2

EX:

```
SELECT  
COUNT(CustomerID)  
FROM Orders  
GROUP BY OrderID  
HAVING  
COUNT(CustomerID) > 2  
ORDER BY COUNT(CustomerID) DESC;
```

ITM 500: Data & Information Management

10260	90	5	1997-08-02	1
-------	----	---	------------	---

ITM 500: Data & Information Management