



SQL knowledge for CloudSAMS

- DAT module
- Basic SQL knowledge (30 mins)
- Learning Query by Examples (40 mins)
- Break (15 mins)
- Joining tables (40 mins)
- Exercise (40 mins)
- Optional Part
 - More SQL examples
 - Tools for Sybase database
 - Interactive SQL
 - Sybase Central

Training material download

All PowerPoint, exercises and suggested answers
can be downloaded at :

<https://cdrcloudsams.edb.gov.hk/>



雲端校管系統資料庫
CloudSAMS Central Document Repository



版本升級

系統保安及系統事宜

雲端服務

模組資料

培訓課程

資料調查

聯絡我們



培訓課程



The screenshot displays the CloudSAMS Central Document Repository website. On the left is a blue sidebar with the CloudSAMS logo and version information (Version 4.0.0a1, 版本 4.0.0a1). A red box highlights the text "CloudSAMS Central Document Repository" and "雲端校管系統資料庫". The main content area features a login section with fields for User ID and Password, and a "Login 登入" button. Below the login section are links for "Forget Password? 忘記密碼?" and "Login via Common Log-On System (CLO) 經統一登入系統登入". To the right of the login section are four green tabs: "覽", "課程講義", "學員須知", and "行事備忘". Below these tabs is a list of training materials, each with a red PDF icon and a blue PowerPoint icon, followed by download links labeled "中" and "下". A red arrow points from the "培訓課程" link in the top navigation bar to the "課程講義" tab.

Training Material	Download Link (Red)	Download Link (Blue)
措施簡介會	中 下	中 下
全年工作流程 (中學適用)	中 下	中 下
全年工作流程 (小學適用)	中 下	中 下
中學)	中 下	中 下
小學)	中 下	中 下
獨立版時間表編排工具簡介會	中 下	中 下

Training material download (cont'd)

培訓課程

主頁 > 培訓課程

課程訊息

課程概覽

課程講義

學員須知

行事備忘

(五)結構性查詢語言(SQL)工作坊

結構性查詢語言[SQL] 初階(一)

中 下載

中 下載

結構性查詢語言[SQL] 初階(二)

中 下載

中 下載

結構性查詢語言[SQL] 進階

中 下載

中 下載

■ Data Management > Query Maintenance

[S-DAT03-03] Data Management > Query Maintenance

▼ Modify Advance Query

✓ Save Go To Execute < Back

Query Name	Query Description
tb_stu_student	tb_stu_student

SQL Statement

```
select * from tb_stu_student
```

■ Data Management > Report

> HKAT

> Applied Learning

> HKEAA

> FMP

> SPA

> WFSFAA(SFO)

> Report Management

▼ Data Management

Query Maintenance

Query Sharing

Task Maintenance

Table Access Control

Execution

Export

Import

Report

Extract for SOP

[S-DAT08-02] Data Management > Report > Table Structure

ER Diagram

Table Structure

Data Manual

Query List

Search

Clear

Table Name

TB_STU_STUDENT

Table Description

Field Name

Field Description

Module

All

<<

<

1

page of 1

>

>>

Display

20

records / page

Table Name ▲	Table Description ⇅
TB_STU_STUDENT	Student

Basic SQL knowledge for CloudSAMS

- Target audience:
- School CloudSAMS users who are not familiar with SQL.

What is SQL?

- SQL stands for **S**tructured **Q**uery **L**anguage.
- SQL allows you to access a RDBMS (Relational Database Management System)
- It presents data as a collection of rows and columns
- SQL has three major parts :
 - DDL – Data Definition Language
 - DML – Data Manipulation Language
 - DCL – Data Control Language

Only **select** is
allowed in
CloudSAMS!

DML:

i) Syntax for executing
queries

ii) e.g. **SELECT** , **INSERT** ,
UPDATE , **DELETE**

iii) SQL Example:

```
select * from  
TB_STU_STUDENT
```

DDL:

i) Create, change and delete
database object (e.g. tables,
indexes, views)

ii) Command Keywords:
CREATE, **DROP**, **ALTER**

iii) SQL example:

```
CREATE TABLE class (class_no  
INTEGER, stud_name CHAR(40))
```

DCL:

Control access right

e.g. **GRANT**

Terms of Relational Database

- Tables
- Fields
- Records
- Keys
- Views
- Indexes

- A database contains many tables
- table is a set of data elements (values) that is organized using a model of horizontal rows and vertical columns.
- The logical order of records and fields within a table is of absolutely no important.
- Example:

- A field is the smallest structure in the database.
- Fields are the structures that are actually used to store data.
- Field name – not case sensitive

- A record represents a unique instance of the subject of a table.
- It is composed of the entire set of fields in a table, regardless of whether or not the fields contain any values.

Table:

Fields (columns)

Records
(rows)

*School ID	*Schyear	*Class	*Class no	Student name	Sex	...
1293	2006	1A	1	John	M
1293	2006	1B	2	Mary	F
1293	2006	1C	3	Ken	M

- A key field is a field of a database table which together form a unique identifier for a database record.
- The aggregate of these fields is usually referred to simply as "the key".
- Example:
 - door key > locate a house
 - schyear,class,classno > locate a student
- Primary key: A Primary Key is a field (or a group of fields) that can be used to identify a unique row in a table.
- Foreign key: A Foreign Key in one table refers to the primary key of another table.
- For example, I write an email to introduce a book to you, with ISBN of the book (not the entire text of the book). The ISBN is the primary-key of the book, and it is used as a foreign-key in the e-mail.

- An index is a feature in a database that allows quick access to the rows in a table.
- The index is created using one or more columns of the table.
- The index is optimized for quick searching
- Example:
 - If a magazine (without table of content) is a table,
 - the table of content is a kind of index
 - The 'magazine' contains fields of page no, subject titles, contents
 - The 'table of contents' contains fields of page no and subject titles

Table and Index

Table:

Fields (columns)

Records
(rows)

*School ID	*Schyear	*Class	*Class no	Student name	Sex	...
1293	2016	1A	1	John	M
1293	2016	1B	2	Mary	F
1293	2016	1C	3	Ken	M

SQL :
create index

Index:

*School ID	*Schyear	*Class	*Class no
1293	2016	1A	1
1293	2016	1B	2
1293	2016	1C	3

* = Keys

Example: (table in CloudSAMS)

Table: TB_STU_STUDENT

Keys: SUID, STUID

SELECT * FROM TB_STU_STUDENT											
Resultset #1 Messages											
	SUID	STUID	AREACODE	BIRTHCERT	CCC	CHBLKNO	CHBUILDING	CHDISTRICT	CHFLATNO	CHFLOORNO	CHNAME
	± 9999	± 1786	3	876WD2		A	舟逸樓	調景嶺	4	25	麥學生
	± 9999	± 1788	3	1052KT1		G	享逸樓	青衣	9	10	吳學生
	± 9999	± 1790	3			K	雍逸樓	將軍澳	2	2	周學生
	± 9999	± 1792	3	Z293261QEH/2		F	東盛樓	荃灣	11	3	劉學生
	± 9999	± 1787	2			K	雍逸樓	青衣	18	27	林學生
	± 9999	± 1789	3			T	清逸樓	觀塘	33	34	陳學生
	± 9999	± 1791	3	Z292660		L	東盛樓	元朗	8	2	朱學生
	± 9999	± 1793	3			B	太盛樓	天水圍	24	31	李學生
	± 9999	± 1798	2			T	樂山樓	將軍澳	31	2	陳學生
	± 9999	± 1794	3	Z364749SPK/1		A	天利樓	青衣	2	37	鄧學生
	± 9999	± 1795	3	Z293642QEH/2		K	舟逸樓	天水圍	25	2	黃學生
	± 9999	± 1796	3	Z200945		H	雍逸樓	青衣	23	43	張學生
	± 9999	± 1797	3	Z403165		G	享逸樓	觀塘	15	23	劉學生
	± 9999	± 1799	3	Z461KUUPRI		E	安逸樓	元朗	25	3	馮學生
	± 9999	± 1800	3	Z10QEH2		E	清逸樓	沙田	4	31	吳學生
	± 9999	± 1801	3	Z074QEH2		J	至逸樓	荃灣	18	37	郭學生
	± 9999	± 1802	3			B	至逸樓	長沙灣	26	2	朱學生
	± 9999	± 1803	3			B	清逸樓	調景嶺	17	34	江學生
	± 9999	± 1804	3			F	雍逸樓	青衣	27	2	陳學生
	± 9999	± 1805	3	Z510SPK1		D	東盛樓	觀塘	22	25	李學生

Example: (table in CloudSAMS)

Table: TB_SCH_SCHCLASS

Keys: SUID, SCHYEAR, SCHLEVEL, SCHSESSION, CLASSCODE

SELECT * FROM TB_SCH_SCHCLASS

Resultset #1 Messages

	SUID	SCHYEAR	SCHLEVEL	SCHSESSION	CLASSLEVEL	CLASSCODE	CLASSNAME	LANGUAGE TYPE	STREAM	FLO
	9999	1988	3	3	S1	1A	1A	1	8	0
	9999	1988	3	3	S1	1C	1C	1	8	0
	9999	1989	3	3	S1	1A	1A	1	8	0
	9999	1989	3	3	S1	1B	1B	1	8	0
	9999	1989	3	3	S1	1C	1C	1	8	0
	9999	1989	3	3	S1	1D	1D	1	8	0
	9999	1989	3	3	S1	1E	1E	1	8	0
	9999	1989	3	3	S1	1F	1F	1	8	0
	9999	1989	3	3	S2	2A	2A	1	8	0
	9999	1989	3	3	S2	2C	2C	1	8	0
	9999	1990	3	3	S1	1A	1A	1	8	0
	9999	1990	3	3	S1	1B	1B	1		0
	9999	1990	3	3	S1	1C	1C	1	8	0
	9999	1990	3	3	S1	1D	1D	1	8	0
	9999	1990	3	3	S1	1E	1E	1	8	0
	9999	1990	3	3	S2	2A	2A	1	8	0
	9999	1990	3	3	S2	2B	2B	1	8	0
	9999	1990	3	3	S2	2C	2C	1	8	0

1, Pos 29 Conn.: websams (Adaptive Server Anywhere) Execution time: 0:0:0.517

- Table Structure
 - Field description
 - Data type
 - Key
 - Index

[S-DAT08-02] Data Management > Report > Table Structure

ER Diagram **Table Structure** Data Manual Query List

Search Clear

Table Name Table Description

Field Name Field Description

Module

<< < 1 page of 1 > >> Display 20 records / page

Table Name	Table Description
TB_STU_STUDENT	Student

- You can get a report of table structure for difference modules from DAT module.

- Information in the report
 - Table
 - Name & Description (Comment)
 - Field of Record
 - Name, Description (Comment), Data Type, Mandatory, Primary Key & Foreign Key
 - Index
 - Name & Column list

- Character Data Types
 - CHAR(n), VARCHAR(n), LONG VARCHAR
- Numeric Data Types
 - INTEGER, UNSIGNED INT(n), SMALLINT, DECIMAL(n,p), NUMERIC(n,p)
 - FLOAT, DOUBLE
- Date & Time Data Types
 - DATE, TIMESTAMP
- Other Data Types
 - BIT(n), LONG BINARY

- If a field is not mandatory, then the field value can be NULL – a missing or unknown value.
- NULL does not represent a zero, a character string of one or more blank spaces, or a "zero-length" character string.

Unknown Values: NULL

- A NULL in a column means that the user or application has made no entry in that column. A data value for the column is undefined.
- NULL does not mean the same as zero (numerical values) or blank (character values). Rather, NULL values allow you to distinguish between a deliberate entry of zero for numeric columns or blank for character columns and a non-entry, which is NULL for both numeric and character columns.

Learning Query By Example

The SELECT statement

SELECT statement

```
SELECT column_list  
FROM table_list  
[WHERE search_condition]  
[GROUP BY column_list]  
[HAVING search_condition]  
[ORDER BY {column_list | column_index}]
```

} constraints

Keys: [] - optional
 { } - list of parameters
 | - or

Notice:

The result set usually becomes smaller in size
when no. of constraints increase

```
SELECT column_list FROM table_list
```

```
SELECT * FROM TB_STU_Student
```

- * stands for all fields in the table

```
select enname, cname  
from TB_STU_STUDENT
```

```
select distinct classcode  
from TB_STU_STUDENT
```

```
select enname + ' (' + cname + ')'  
from TB_STU_STUDENT
```

```
select
  enname as 'English Name',
  chname 'Chinese Name',
  enname + ' (' + chname + ')' Student
from TB_STU_STUDENT
```

Syntax:

- Column-name AS alias
- Column-name alias

```
select
```

```
  a.ename AS 'English Name',
```

```
  a.chname 'Chinese Name',
```

```
  a.ename + ' (' + a.chname + ')' Student
```

```
from TB_STU_STUDENT a
```

ORDER BY clause

```
select distinct classcode  
from TB_STU_STUDENT  
order by classcode
```



```
select classcode, classno, enname, chname  
from TB_STU_STUDENT  
order by classcode desc, classno asc
```



```
select classcode, classno, enname, chname  
from TB_STU_STUDENT  
order by 1 desc, 2 asc
```



Exercise 1 (order by)

Use table TB_STU_STUDENT ,
find student class, classno, name
Sort the result set by class and class no.

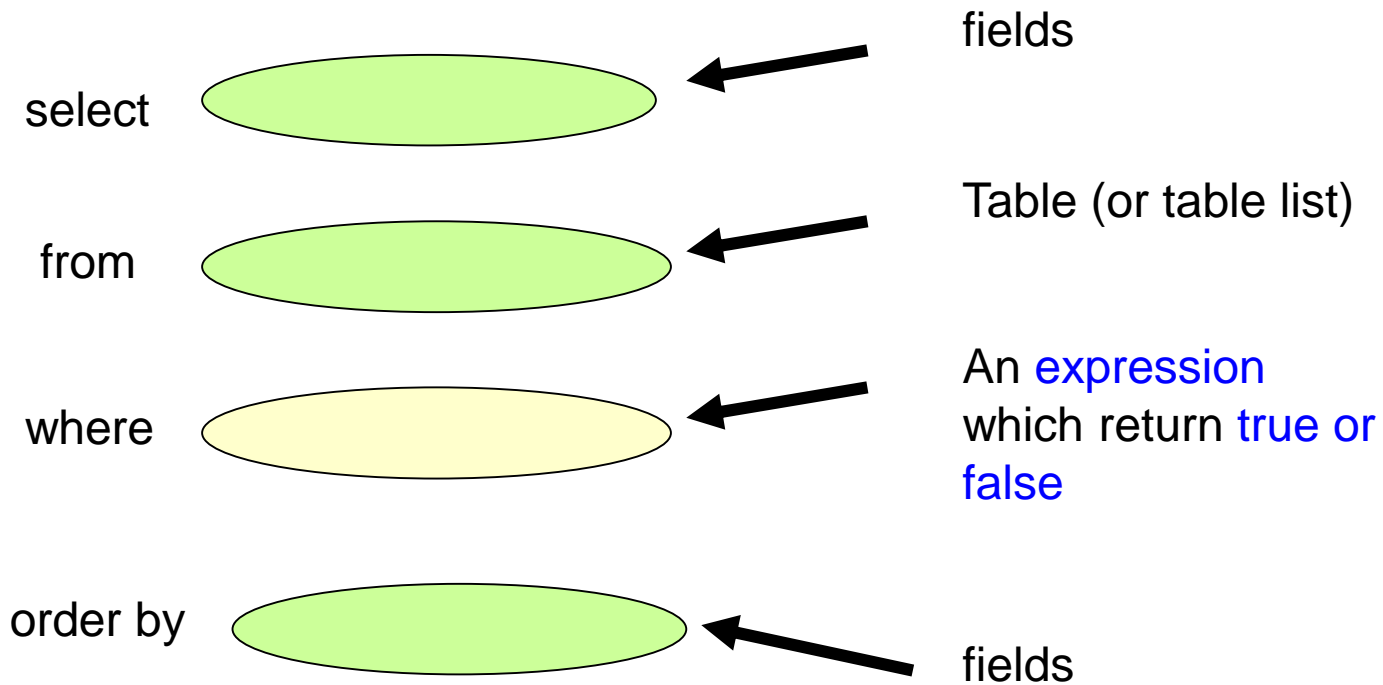
Hints: How to check the field name? Go to Data Management > Report

Suggested answers:

```
select classcode, classno, enname, chname  
from TB_STU_STUDENT  
order by classcode, classno
```

WHERE clause with one expression

- [WHERE search_condition]




```
select classcode, classno, enname, chname  
from TB_STU_STUDENT  
where classlvl = 'S1'  
order by classcode, classno
```

Search Condition for WHERE

- Comparison (=, <, >, <=, >=, <>)
- Range
 - [NOT] BETWEEN start_value AND end_value
- Membership
 - [NOT] IN (value list)
- Pattern Match
 - [NOT] LIKE pattern_string
 - Wildcard characters : % _
- NULL
 - IS [NOT] NULL

Sample Queries

```
SELECT ClassCode, EnName, ChName  
FROM TB_STU_Student  
WHERE ClassCode BETWEEN '4' AND '5Z'  
ORDER BY ClassCode, EnName
```

```
SELECT ClassCode, EnName, ChName  
FROM TB_STU_Student  
WHERE ClassLvl IN ('S4', 'S5')  
ORDER BY ClassCode, EnName
```



Sample Queries

```
SELECT ClassCode, EnName, ChName  
FROM TB_STU_Student  
WHERE EnName LIKE 'CHAN %'  
ORDER BY EnName
```



```
SELECT ClassCode, EnName, ChName  
FROM TB_STU_Student  
WHERE HKID IS NULL  
ORDER BY ClassCode, EnName
```



Simple logic for where condition

```
select ....  
from ....  
where condition1 or condition2
```

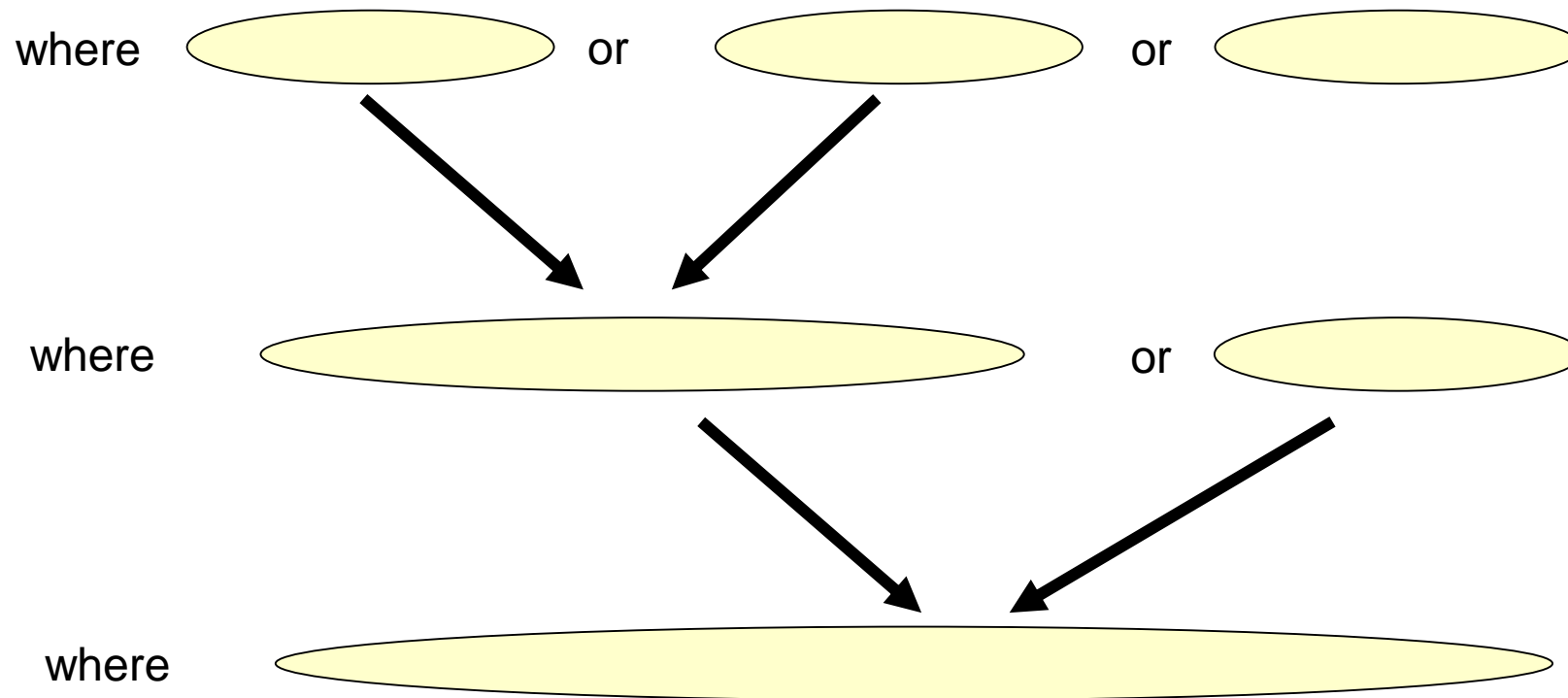
```
select ....  
from ....  
where condition1 and condition2
```

T or T = True
F or F = False
T or F = True
F or T = True

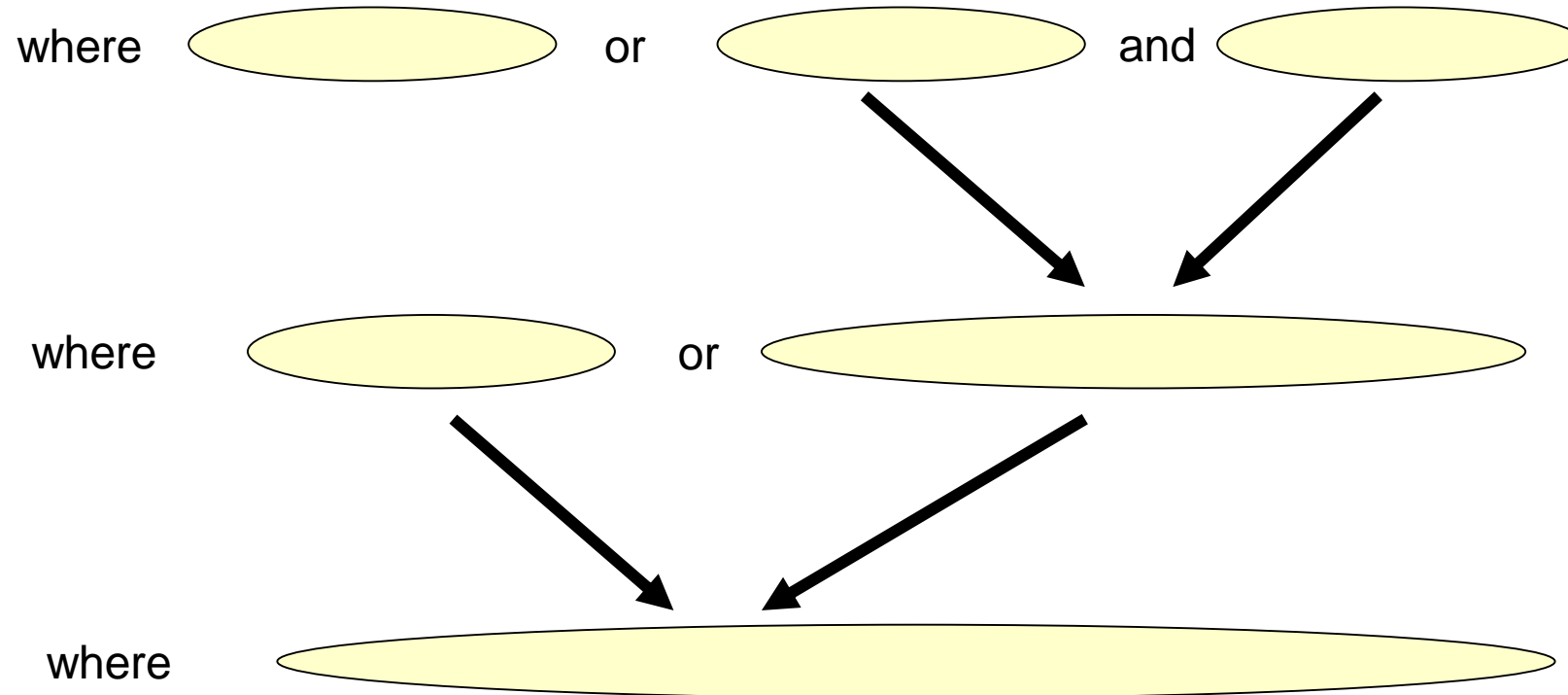
T and T = True
F and F = False
T and F = False
F and T = False

WHERE clause with two or more expression

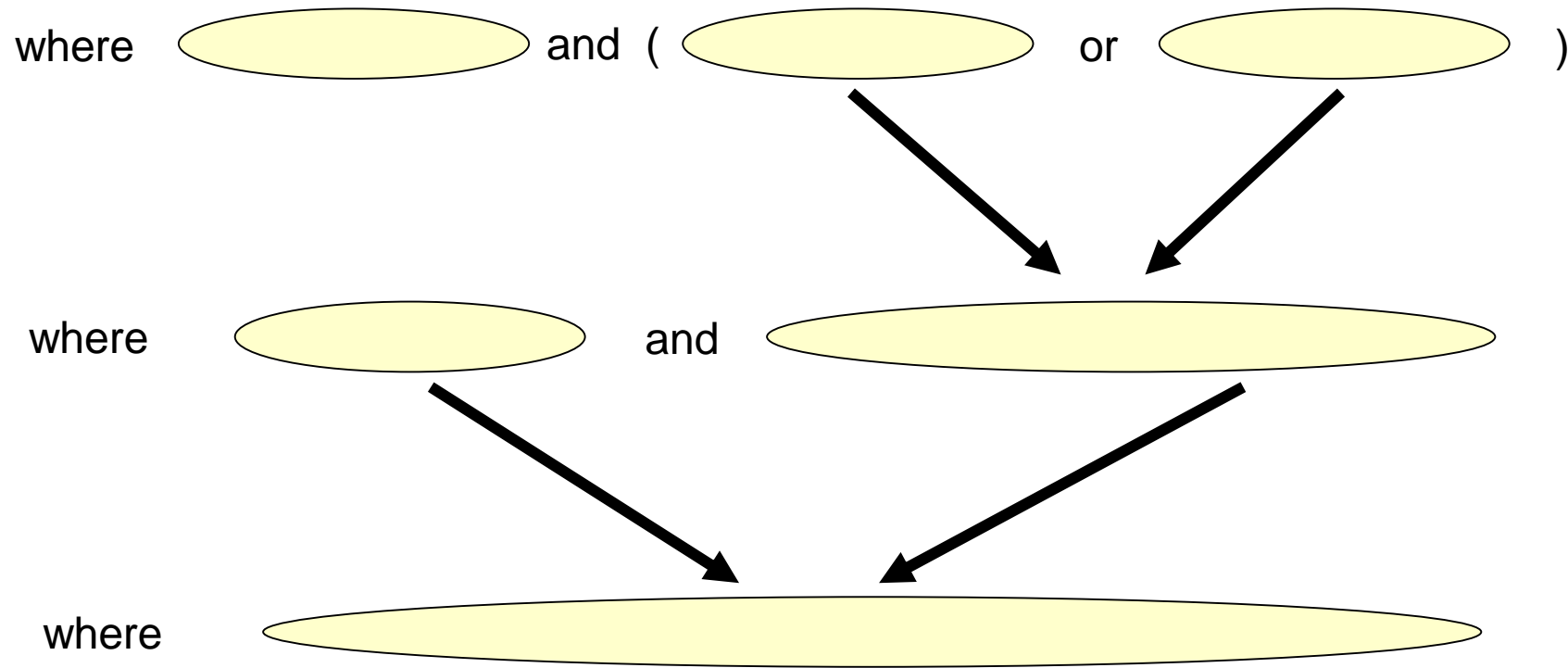
Reminder: each expression will only returns true or false.



WHERE clause with two or more expression (cont'd)



WHERE clause with two or more expression (cont'd)



Sample Queries

```
SELECT ClassCode, EnName, ChName  
FROM TB_STU_Student  
WHERE  
    ClassLv IN ('S4', 'S5')  
AND  
    EnName LIKE 'CHAN %'  
ORDER BY ClassCode, EnName
```

True or False?

True or False?



Sample Queries

```
SELECT ClassCode, ClassNo, EnName, ChName, STRN, HKID
FROM TB_STU_Student
WHERE
  ( HKID IS NULL OR STRN IS NULL )
AND
  ClassCode BETWEEN '1' AND '2Z'
ORDER BY ClassCode, ClassNo
```

True or False?

True or False?

True or False?



Useful Functions (Date)

- YEAR(date_field)
- MONTH(date_field)
- DAY(date_field)
- NOW()
- getDate()
- DateFormat(date_field, date_format)
- E.g. `select dateformat(date_field, 'DD/MM/YYYY') from`

```
SELECT  
    DOB,  
    YEAR(DOB),  
    MONTH(DOB),  
    DAY(DOB),  
    NOW(),  
    DATEFORMAT(DOB, 'DD/MM-YYYY')  
FROM TB_STU_Student
```

Notice:

A field can be selected more than once,
depends on the data set you need



Useful Functions (String)

- UPPER(text_field)
- LOWER(text_field)
- TRIM(text_field)
- LEFT(text_field, length)
- RIGHT(text_field, length)
- SUBSTRING(text_field, start position, length)
- LENGTH(text_field)

Sample Queries

```
SELECT  
    EnName,  
    LOWER(EnName),  
    TRIM(EnName),  
    SUBSTRING(EnName, 3, 5),  
    LENGTH(EnName)  
FROM TB_STU_Student
```



Exercise 2 (substring)

Find out the classcode, class no. , student name, first 6 digits of strn of each student, order by class and class no., and format the student name as , e.g. Chan Tai Man (陳大文)

Hints: How to check the field name? Go to Data Mgmt > Report

Use function: substring(text_field, start position, length)

Syntax: text_field + ' (' + text_field2 + ') '

Suggested Answer:

```
select classcode, classno, enname + ' (' + chname + ' ) 'first 6 digits'
from TB_STU_STUDENT
order by classcode, classno
```

Exercise 3 (where)

Also display the 'date of birth' field, with format DD/MM/YYYY
Filter the students with birth year 2006,
and the class code of the student should not be empty.

Hints:

Use function: `dateformat(date_field, 'DD/MM/YYYY')`
`Year(field_name)`

Suggested answers:

```
select classcode, classno, ename + ' ('+chname+)' , substring( strn, 2, 6 ) 'first 6 digits' , dateformat( dob,
'DD/MM/YYYY')
from TB_STU_STUDENT
where YEAR(dob) = ? and classcode <> ''
order by classcode, classno
```


Aggregate Functions

- COUNT([DISTINCT] field)
- SUM([DISTINCT] field)
- AVG([DISTINCT] field)
- MAX(field)
- MIN(field)

GROUP BY clause

- [GROUP BY column_list]
- When you use aggregate functions in the SELECT clause to produce summary information, you use the GROUP BY clause to divide the information into distinct groups.
- The GROUP BY clause is optional.

What is “Count” function?

```
select ClassCode ,EnName
from TB_STU_Student
```

ClassCode	EnName
1A	Mary
1A	John
1A	Ken
1B	David
1B	Peter
1C	Calvin

```
select ClassCode , COUNT(EnName)
from TB_STU_Student
group by ClassCode
```

ClassCode	Count(EnNam)
1A	3
1B	2
1C	1

Sample Queries 1

This SQL find out the number of student in each class:

```
select  
  ClassCode 'Class',  
  COUNT(EnName) 'No. of Student'  
from TB_STU_Student  
group by ClassCode  
order by ClassCode
```

Think: a SQL has count(..) but no 'group by' ...
e.g. select count(*) from TB_STU_STUDENT



Sample Queries 2

This SQL find out the number of student in Form 1, group by class and gender:

```
SELECT  
  ClassCode 'Class',  Sex, COUNT(*) 'No. of Student'  
FROM TB_STU_Student  
WHERE ClassCode in ('1A','1B','1C','1D','1E')  
GROUP BY ClassCode, Sex  
ORDER BY ClassCode, Sex
```



Sample Query 3

The table TB_STU_STUSCHREC stores all the schooling record of each student in each year.

```
select stuid,  
MIN( firstattdate ) 'First Attend Date'  
from TB_STU_STUSCHREC  
group by stuid
```

stuid	firstattdate	Class
0001	1-9-2006	1A
0002	1-9-2005	1A
0002	31-9-2006	1B
0003	1-9-2004	2E
0003	12-9-2005	2E
0003	1-9-2006	2B
0004	1-9-2004	2B
0004	1-9-2005	2B
0005	1-9-2006	5A
0006	1-9-2006	5B



Exercise 4 (function)

- 1) Use table TB_STU_STUDENT, find how many student in the class 1A, 2B 3C and 4D respectively.

Hint: use the function *count*(field_name)
use fields classcode and ename

Suggested answers:

```
select classcode, count( ename )  
from TB_STU_STUDENT  
where classcode in ( '1A', '2B' , '3C' , '4D' )  
group by classcode  
order by classcode
```

- 2) Use table TB_STU_STUSCHREC, find the first attend date of each student.

Hint: use function *min*(field_name), and fields stuid, firstattdat

Suggested answers:

```
select stuid, min(firstattdat) 'First Attend Date'  
from TB_STU_STUSCHREC  
group by stuid
```

HAVING clause

- [HAVING search_condition]
- The HAVING clause is specifically associated with the GROUP BY clause, and you use it to filter the grouped information.
- *It is similar to the WHERE clause in that the HAVING keyword is followed by an expression that evaluates to TRUE, FALSE or UNKNOWN.
- HAVING is also an optional clause.

Sample Queries 1

```
SELECT
  ClassCode 'Class',
  COUNT(EnName) 'No. of Student'
FROM TB_STU_Student
WHERE ClassCode <> ''
GROUP BY ClassCode
HAVING COUNT(EnName) > 40
ORDER BY ClassCode
```

Note:

- 1) 'HAVING' is needed only when 'GROUP BY' exists.
- 2) This is a wrong syntax:
...where COUNT(EnName) > 40



Exercise 5 (having)

5.1) Continue Ex.4.1, find out the class with more than 40 students within the class 1A, 2B and 3C and 4D :

Suggested answers:

```
select classcode, count( ename ) from TB_STU_STUDENT
where classcode in ( '1A', '2B' , '3C' , '4D' )
group by classcode
having count( ename ) > 40
order by classcode
```

5.2) Continue Ex.4.2, find out the student who join the school after 2016-09-02:

Hint: First find the minimum 'First Attend Date' for each stuid, then use having for filtering

Use function: datetime('YYYY-MM-DD')

Suggested answers:

```
select stuid, min(firstattdate) 'First Attend Date'
from TB_STU_STUSCHREC
group by stuid
having min(firstattdate) > datetime('2016-09-02')
```

CASE

WHEN condition1

THEN result1

WHEN condition2

THEN result3

....

ELSE

result

END

Sample Queries

```
SELECT  
  ClassCode, ClassNo, AreaCode,  
CASE  
  WHEN AreaCode = 1 THEN 'Hong Kong'  
  WHEN AreaCode = 2 THEN 'Kowloon'  
  WHEN AreaCode = 3 THEN 'N.T.'  
  ELSE 'N/A'  
END 'AREA'  
FROM TB_STU_Student  
ORDER BY ClassCode DESC, ClassNo
```



Exercise 6.1 (case when)

From subject assessment table (TB_ASR_SUBJASSESSDATA),
Find student id (stuid), subjcode (subjcode), and score (sysscore).
For code '135', display 'economics',
for code '165', display 'english', others display 'Others'
School year and time sequence should be 2023 and 1100 respectively.

SYNTAX:

```
CASE  
  WHEN .... THEN ....  
  WHEN .... THEN ....  
  ELSE ....  
END
```

Modify this SQL:

```
select  
stuid,  
subjcode,  
subjcode 'Subject Name',  
sysscore  
from TB_ASR_SUBJASSESSDATA  
where schyear=2023  
and timeseq=1100
```

Suggested answers:

```
select stuid, subjcode,  
case  
  when subjcode='135' then 'economics'  
  when subjcode='165' then 'english'  
  else 'Others'  
end 'Subj name',  
sysscore  
from TB_ASR_SUBJASSESSDATA  
where schyear=2023 and timeseq=1100
```

Exercise 6.2 (case when) DEMO

From parent table (TB_STU_PARENT),
find the name of parents, and the relation value.

For the relation value, display 'mother' for '01' , 'father' for '02', 'other' for other values.

Suggested Answer:

```
select ename, cname,  
case  
when relation='01' then 'mother'  
when relation='02' then 'father'  
else 'other'  
end 'relation'  
from TB_STU_PARENT
```

SELECT column_list

FROM table_list

[WHERE search_condition]

[GROUP BY column_list]

[HAVING search_condition]

[ORDER BY {column_list | column_index}]

} constraints

Keys: [] - optional
{ } - list of parameters
| - or

Writing Practical Query for CloudSAMS

Advanced query syntax

- Join table
 - Left outer join, inner join
- Sub-query
 - Field and search condition
- Combining query

- Inner join
- Left Outer join

Join Table

■ TB_NAME

*stuid	name
001	Ken
002	May
003	John

■ TB_CLASS

*stuid	classcode	classno
001	4A	18
002	2E	5
004	1C	26

Inner Join

- An Inner Join returns only those rows where the linking values(keys) match in both of the tables or in Result Sets

Example:

Table alias: refer the field to table a and b

```
select a.stuid , a.name, b.classcode, b.classno  
from TB_NAME a  
join TB_CLASS b  
on a.stuid = b.stuid
```

Example of Inner Join

Alternative version:

```
select a.stuid, a.firstname, b.classcode, b.classno  
from TB_NAME a, TB_CLASS b  
where a.stuid=b.stuid
```

Result:

a.stuid	a.name	b.classcode	b.classno
001	Ken	4A	18
002	May	2E	2

Exercise 7.1 (inner join)

抽取全校平均分獲70分的學生名單

Table: **VW_STU_LATESTSTUDENT** and **TB_ASR_STUDASSESSDATA** by using inner join.

學生成績資料

學生資料

suid	stuid	schyear

a.suid=b.suid

a.stuid=b.stuid

a.schyear=b.schyear

?

Inner join

suid	stuid	schyear	syspercscscore

Result set

classlevel	classcode	classno	Student name	syspercscscore

Exercise 7.1 cont'd (inner join)

抽取2023學年Term 1全校平均分獲70分或以上的學生名單

Table:

VW_STU_LATESTSTUDENT and TB_ASR_STUDASSESSDATA by using inner join.

Suggested answer:

Select

a.classlvl, a.classcode, a.classno, a.chname,b.syspercscscore 'average'

from VW_STU_LATESTSTUDENT a
join TB_ASR_STUDASSESSDATA b
on a.suid = b.suid and a.STUID = b.STUID
and a.schyear = b.schyear

where b.schyear = ? and b.TIMESEQ = ? and b.syspercscscore >= 70
order by a.classlvl,a.classcode, a.classno

參數例子： TIMESEQ -- 1000 (全年), 1100 (T1), 1201 (T2A1), 1302 (T3A2)
SYSPERCSCORE -- 平均分下限

Code Table Mapping

- Common code table

Most frequently used: TB_HSE_COMMON

- Independent code table

- Class Level (TB_HSE_ClsLvl)
- Qualification (TB_HSE_QualCD)
- Subject Component (TB_HSE_SbjCmp)
- Training Course (TB_HSE_TrnCD)

Code Table Mapping – TB_ID

[S-DAT08-02] Data Management > Report > Table Structure

ER Diagram
Table Structure
Data Manual
Query List

Search

Table Name

Table Description

Field Name

Field Description

Module
Code Management

<< < 1 page of 1 > >> Display 20 records / page

Table Name ^	Table Description ⇅
TB_HSE_ACT	History of Action
TB_HSE_CLSLVL	Class Level Code
TB_HSE_COMMON	Common Code
TB_HSE_CTABLE	Code Table
TB_HSE_HIST	History of Changes
TB_HSE_HLDAYEVT	Table for storing subject mapping history
TB_HSE_LIMIT	Limit Table
TB_HSE_MODFY	History of Modification
TB_HSE_PARAMETER	This table contains the pre-define parameters using in WebSAMS
TB_HSE_QUALCD	Qualification Code
TB_HSE_RCHRRELAT	Rank Change Relationship
TB_HSE_RTYPRELAT	Rank Type Relationship
TB_HSE_SBJCMP	Subject Component Code

Use code table, find subject name

Step 1) Subject Assessment Table:

1st term score in 2023:

```
select schyear, suid, stuid, subjcode,
sysscore
from TB_ASR_SUBJASSESSDATA
where schyear=2023 and timeseq=1100
```

schyear	suid	stuid	subjcode	sysscore
2006	9999	250	076	33.00
2006	9999	250	085	45.00
2006	9999	250	091	23.00
2006	9999	250	185	
2006	9999	250	265	32.00
2006	9999	250	310	
2006	9999	250	430	32.00
2006	9999	394	071	38.00
2006	9999	394	085	38.00

TB_ID = 'SBJ'

Step 2) Code table:

```
select suid, tb_id, code_id, en_des
from TB_HSE_COMMON
where tb_id = 'SBJ'
```

SUID	TB_ID	CODE_ID	EN_DES	CH_DES
9999	RELATE	01	Father	父親
9999	RELATE	02	Mother	母親
9999	SBJ	085	Chinese Language and Culture	中國語文及文化
9999	SBJ	090	Chinese Literature	中國文學
9999	SBJ	091	Chinese Literature(AL)	中國文學(高級)
9999	SBJ	095	Civic Education (S1-3)	公民教育(中一至中三)
9999	SBJ	100	Commerce	商業

TB_HSE_COMMON (b)

suid	tb_id	Code_id	Description
9999	RELATE	01	Mother
9999	RELATE	02	Father
9999	SBJ	076	History
9999	SBJ	085	Chinese
9999	ECACD	1010	Drama Club

TB_ASR_SUBJASSESSDATA (a)

suid	subjcode	score
9999	076	33
9999	085	45
9999

a.suid → b.suid

b.tb_id → 'SBJ'

a.subjcode → b.code_id

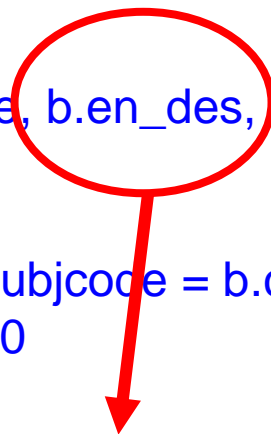
INNER JOIN
(values match both table)

Result set

a.suid	a.Stuid	a.subjcode	b.description	a.score
9999	250	076	History	33
9999	394	085	Chinese	45

Step 3) Combine the SQLs by inner join:

```
select a.schyear, a.suid, a.stuid, a.subjcode, b.en_des, a.sysscore
from TB_ASR_SUBJASSESSDATA a
join TB_HSE_COMMON b
on a.suid=b.suid and b.tb_id='SBJ' and a.subjcode = b.code_id
where a.schyear=2023 and a.timeseq=1100
```




schyear	suid	stuid	subjcode	en_des	sysscore
2016	9999	1213	075	Chinese History	68.00
2016	9999	1213	080	Chinese Language	122.00
2016	9999	1213	185	Ethics/ Religious Education	19.00
2016	9999	1213	310	Physical Education Lessons	20.00
2016	9999	1213	350	Putonghua	17.00
2016	9999	1213	035	Art And Design	19.00
2016	9999	1213	110	Computer Literacy	20.00
2016	9999	1213	130	Design And Technology	20.00

Exercise 7.2 (common code table)

Exercise: Find the parent name and translate the 'relation' code of each parent by the common code table:

Step 1) Parent table:


```
SELECT suid, relation, ename 'Parent
Name', Relation 'Code'
FROM TB_STU_Parent
ORDER BY EnName
```



suid	Relation	Parent Name
9999	01	PARENT 1951
9999	02	PARENT 2251
9999	02	PARENT 2313
9999	02	PARENT 2313

Step 2) Code table:

```
select suid, tb_id, code_id, en_des
from TB_HSE_COMMON
where tb_id = 'RELATE'
```



SUID	TB_ID	CODE_ID	EN_DES	CH_DES	C
9999	RELATE	01	Father	父親	A
9999	RELATE	02	Mother	母親	A
9999	RELATE	03	Grandfather	祖父	A
9999	RELATE	04	Grandmother	祖母	A
9999	RELATE	05	Brother	兄弟	A
9999	RELATE	06	Sister	姊妹	A

Exercise 7.2 (cont'd)

Suggested Answer:

```
SELECT a.EnName 'Parent Name',  
a.Relation 'Code', b.En_Des 'Relation'  
FROM TB_STU_Parent a  
JOIN TB_HSE_Common b  
on b.SUID = a.SUID  
and b.Tb_ID = 'RELATE'  
and b.Code_ID = a.Relation  
order by a.EnName
```



Left Outer Join

- A **left outer join** is very different from an inner join. Instead of limiting results to those in both tables, it limits results to those in the "left" table (A). This means that if the ON clause matches 0 records in B, a row in the result will still be returned—but with NULL values for each column from B.
- It returns **all the values from left table** + only matched values from right table.

Left Outer Join (Example)

```
select a.stuid, a.name, b.classcode, b.classno
from TB_NAME a
left outer join TB_CLASS b
on a.stuid = b.stuid
```

TB_NAME

001	Ken
002	May
003	John

TB_CLASS

001	4A	18
002	2E	5
004	1C	26

TB_CLASS

001	4A	18
002	2E	5
004	1C	26

TB_CLASS

001	4A	18
002	2E	5
004	1C	26

Result :

a.stuid	a.name	b.classcode	b.classno
001	Ken	4A	18
002	May	2E	2
003	John	(NULL)	(NULL)

Left outer join: left table as “master table” , and join the right table

Sample Queries

Getting the parent name of each student:

This returns the name of each student, whenever he/she has a parent record or not

```
select
  a.EnName 'Student Name',
  b.EnName 'Parent Name'
from TB_STU_Student a
  left outer join TB_STU_Parent b
    on
      a.StuID = b.StuID and
      a.SUID = b.SUID
ORDER BY a.EnName
```



Example (Left outer join)

Get Student and Parent Name,
order by class and class no.
from tables TB_STU_Student and TB_STU_Parent

TB_STU_Student

suid	stuid	classcode	classno	enname
9999	1759	1A	1	Chan Cheung Po
9999	1444	1A	2	Chan Fun Keung
9999	1773	1A	3	Chan Ha Wah
9999	1589	1A	4	Chan Ling Don
9999	1645	1A	5	Chan Lu Fun
9999	1607	1A	6	Chan Mei Man

TB_STU_Parent

suid	stuid	enname
9999	92	PARENT0001
9999	93	PARENT0003
9999	94	PARENT0005
9999	95	PARENT0007

LEFT OUTER JOIN

Example

Get Student and Parent Name,
Order by class and class no.:

```
SELECT
  a.ClassCode 'Class Code',
  a.ClassNo 'Class No',
  a.EnName 'Student Name',
  b.EnName 'Parent Name'
FROM TB_STU_Student a
LEFT OUTER JOIN
TB_STU_Parent b
ON
  b.StuID = a.StuID and
  b.SUID = a.SUID
WHERE
  a.ClassCode <> ''
ORDER BY a.ClassCode, a.ClassNo
```

SUID	STUID	...	SUID	STUID	Parent	...

LEFT OUTER JOIN

Class code	Class No	Stud Name	Parent Name
1A	1	Peter	Mr. Wong
1A	2	John	(NULL)
1B	1	Mary	Mr. Chang
1B	2	Tim	Mr. Lee
1B	3	Joe	(NULL)
1C	1	Jack	Mr. Yan
1C	2	Henry	Mr. Lau
1C	3	Eric	Mr. Kwan

Exercise 8.1 (left outer join)

抽取2023學年, 中一級學生Term1考試的中文科分數和級名次

Table: VW_STU_LATESTSTUDENT and
TB_ASR_SUBJASSESSDATA

Use left outer join join.

VW_STU_LatestStudent

suid	stuid	schyear	ClassCode	ClassNo	EnName
9999	1184	2006	3D	2	STUDENT AHAH
9999	1188	2006	3D	3	STUDENT AHBB
9999	1189	2006	3E	3	STUDENT AHBC
9999	1208	2006	3D	7	STUDENT AHEC
9999	1236	2006	3C	9	STUDENT AHHI

TB_ASR_SUBJASSESSDATA

suid	stuid	schyear	sysscore	OMclasslvl
8886	270	2006	45.00	16
8886	3640	2006	53.50	15
8886	3645	2006	66.00	7
8886	3648	2006	89.00	1

Left outer
join

Exercise 8.1 (cont'd)

Suggested answer:

```
select
```

```
  a.schyear ,  a.classlvl ,  a.classcode ,  a.classno ,  a.chname ,  
  b.sysscore ,  b.OMclasslvl
```

```
from VW_STU_LATESTSTUDENT a
```

```
left outer join TB_ASR_SUBJASSESSDATA b
```

```
on a.suid = b.suid and a.stuid = b.stuid and a.schyear = b.schyear and b.subjcode='080' and b.timeseq =  
1100
```

```
where a.schyear = ? and a.classlvl = 'S1'
```

```
order by a.classcode, a.classno
```

Exercise 8.2

再抽取2023學年中一級學生Term1考試的英文科成績

Hints: subject code = '165'

Suggested answer:

select

a.schyear , a.classlvl , a.classcode , a.classno , a.chname ,
b.sysscore 'chi subj score', b.OMclasslvl 'chi OM',
c.sysscore 'eng subj score', c.OMclasslvl 'eng OM'

from VW_STU_LATESTSTUDENT a

left outer join TB_ASR_SUBJASSESSDATA b

on a.suid = b.suid and a.stuid = b.stuid and a.schyear = b.schyear and b.subjcode='080' and b.timeseq = 1100

left outer join TB_ASR_SUBJASSESSDATA c

on a.suid = c.suid and a.stuid = c.stuid and a.schyear = c.schyear and c.subjcode='165' and c.timeseq = 1100

where a.schyear = ? and a.classlvl = 'S1'

order by a.classcode, a.classno

Advanced Topics

- A subquery is an SQL SELECT statement that's nested inside of another SQL statement
- A subquery can appear in the field list (as in the preceding example) or in a WHERE or HAVING clause

Required Query

SUID	STUID	...

SUID	STUID	Parent
		...

LEFT OUTER JOIN

Student without parent record:

```

SELECT
  a.ClassCode 'Class Code',
  a.ClassNo 'Class No',
  a.EnName 'Student Name'
FROM TB_STU_Student a
LEFT OUTER JOIN TB_STU_Parent b
ON
  b.StuID = a.StuID and
  b.SUID = a.SUID
WHERE
  a.ClassCode <> " AND
  b.EnName IS NULL
ORDER BY a.ClassCode, a.ClassNo
  
```

Class code	Class No	Stud Name	Parent Name
1A	1	Peter	Mr. Wong
1A	2	John	(NULL)
1B	1	Mary	Mr. Chang
1B	2	Tim	Mr. Lee
1B	3	Joe	(NULL)
1C	1	Jack	Mr. Yan
1C	2	Henry	Mr. Lau
1C	3	Eric	Mr. Kwan

WHERE

Class code	Class No	Stud Name	Parent Name
1A	2	John	(NULL)
1B	3	Joe	(NULL)

Transformed to Sub-Query

- Student without parent record:
- SELECT
- a.STUID ,
- a.ClassCode 'Class Code',
- a.ClassNo 'Class No',
- a.EnName 'Student Name'
- FROM TB_STU_Student a
- where
- a.classcode <> " and
- a.stuid not in
- (select stuid from tb_stu_parent)
- ORDER BY a.ClassCode, a.ClassNo

STUID	Class code	Class No	Stud Name
001	1A	1	Peter
002	1A	2	John
003	1B	1	Mary
004	1B	2	Tim
005	1B	3	Joe
006	1C	1	Jack

WHERE ...
STUID NOT IN

STUID	Parent Name
001	Mr. Wong
003	Mr. Chang
004	Mr. Lee
006	Mr. Yan

STUID	Class code	Class No	Stud Name
002	1A	2	John
005	1B	3	Joe

Evaluation Order

Evaluation Order	Type of Operator
1	Positive Sign (+), Negative Sign (-)
2	Multiplication (*), Division (/)
3	Addition (+), Subtraction (-)
4	=, <, >, <=, >=, <>, BETWEEN, IN, LIKE, IS NULL
5	NOT
6	AND
7	OR

Advanced Example – Calculate Age

```
SELECT ClassCode, EnName, dob,  
       case  
         when (MONTH(dob) > MONTH(NOW()))  
           then YEAR(NOW())-YEAR(dob)-1  
         when ((MONTH(dob) = MONTH(NOW()))  
              and (DAY(dob) > DAY(NOW())))  
           then YEAR(NOW())-YEAR(dob)-1  
         else YEAR(NOW())-YEAR(dob)  
       end 'AGE'  
FROM TB_STU_Student  
WHERE ClassCode <> ''  
ORDER BY ClassCode, EnName
```



Harder Example – Age Statistics

```
SELECT ClassLvl,  
       YEAR(NOW())-YEAR(DOB)-(CASE  
         WHEN (MONTH(DOB)*40+DAY(DOB) >  
              MONTH(NOW()) *40+DAY(NOW()))  
         THEN 1  
         ELSE 0  
       END) 'AGE',  
       COUNT(EnName) 'No. of Student'  
FROM TB_STU_Student  
WHERE ClassCode <> ''  
GROUP BY ClassLvl, AGE  
ORDER BY ClassLvl, AGE
```



Harder Example – Full Address

```
SELECT ClassCode, ClassNo, EnName,  
  (CASE WHEN TRIM( EnFlatNo )<>"  
    THEN 'Flat '+TRIM( EnFlatNo )+', ' ELSE " END)  
+ (CASE WHEN TRIM(EnFloorNo)<>"  
    THEN 'Floor '+TRIM(EnFloorNo)+', ' ELSE " END)  
+ (CASE WHEN TRIM(EnBlkNo)<>"  
    THEN 'Block '+TRIM(EnBlkNo)+', ' ELSE " END)  
+ (CASE WHEN TRIM(EnBuilding)<>"  
    THEN TRIM(EnBuilding)+', ' ELSE " END)  
+ (CASE WHEN TRIM(EnVillageEstate)<>"  
    THEN TRIM(EnVillageEstate)+', ' ELSE " END)
```

```
+ (CASE WHEN TRIM(EnStreet)<>"  
      THEN TRIM(EnStreet)+', ' ELSE " END)  
+ (CASE WHEN TRIM(EnDistrict)<>"  
      THEN TRIM(EnDistrict)+', ' ELSE " END)  
+ (CASE WHEN AreaCode=1 THEN 'Hong Kong'  
      WHEN AreaCode=2 THEN 'Kowloon'  
      WHEN AreaCode=3 THEN 'New Territories'  
      ELSE " END) AS Address  
FROM TB_STU_Student  
ORDER BY EnName
```



End of Workshop