Section 08.3 - Data Definition Language (DDL) and Data Manipulation Language (DML)

```
## Layer 7: Application
```

Syllabus Content Section 08: Databases

S08.3.1 Show understanding that DBMS carries out all creation/modification of the database structure using its Data Definition Language (DDL)

`DDL(Data Definition Language)`: Creates databases and creates alter tables

> Example

```
CREATE DATABASE BandBooking;
CREATE TABLE Band (
        BandName varchar(25),
        NumberOfMembers integer);
ALTER TABLE Band ADD PRIMARY KEY (BandName);
ALTER TABLE Band-Booking ADD FOREIGN KEY (BandName REFERENCES
Band(BandName);
```

S08.3.2 Show understanding that the DBMS carries out all queries and maintenance of data using its DML

`DML(Data Manipulation Language)`: Insert and changes data in your table

> Example

```
INSERT INTO Band (‘ComputerKidz’, 5);
INSERT INTO Band-Booking (BandName, BookingID)
VALUES (‘ComputerKidz’, ‘2016/023’);
```

S08.3.3 Show understanding that the industry standard for both DDL and DML is Structured Query Language (SQL)

- Understand a given SQL script

SELECT	extracts data from a database
UPDATE	updates data in a database
DELETE	deletes data from a database
INSERT INTO	inserts new data into a database
CREATE DATABASE	creates a new database
ALTER DATABASE	modifies a database
ALTER TABLE	modifies a table
CREATE TABLE	creates a new table
DROP TABLE	deletes a table
CREATE INDEX	creates an index (search key)
DROP INDEX	deletes an index

S08.3.4 Understand given SQL (DDL) commands and be able to write simple SQL (DDL) commands using a sub-set of commands

- Create a database (CREATE DATABASE)
- Create a table definition (CREATE TABLE), including the creation of attributes with appropriate data types:
- CHARACTER
- VARCHAR(n)

- BOOLEAN
- INTEGER
- REAL
- DATE
- TIME
- change a table definition (ALTER TABLE)
- add a primary key to a table (PRIMARY KEY (field))
- add a foreign key to a table (FOREIGN KEY (field) REFERENCES Table (Field))

Create a database

```
CREATE DATABASE ;
```

Create a table definition

```
CREATE TABLE <table_name>{
        <column_name> <data_type>,
        <column_name> <data_type>,
        <column_name> <data_type>,
        ......
};
```

Change a table definition

```
/* Add a column: */
ALTER TABLE <table_name>
ADD <column_name> <data_type>;

/* Delete a column: */
ALTER TABLE <table_name>
DROP COLUMN <column_name>;

/* Modify the data type of a column: */
ALTER TABLE <table_name>
ALTER COLUMN <column_name> <data_type>;
```

Add a primary key to a table

```
/* When creating a table: */
CREATE TABLE <table_name>{
        <column_name> <data_type> PRIMARY KEY,
        ......
};

/* MySQL Syntex */
CREATE TABLE <table_name>{
        ......
        PRIMARY KEY(<column_name>)
};

/* When changing a table definition */
ALTER TABLE <table_name>
ADD PRIMARY KEY (<column_name>);
```

Add a froeign key to a table

```
/* When creating a table: */
CREATE TABLE <table_name>{
        <column_name> <data_type> FOREIGN KEY REFERENCES
<table_name>(<column_name>),
        ......
};

/* MySQL Syntex */
CREATE TABLE <table_name>{
        ......
        FOREIGN KEY (<column_name>) REFERENCES <table_name>
(<column_name>)
};

/* When changing a table definition */
ALTER TABLE <table_name>
ADD FOREIGN KEY (<column_name>)
REFERENCES <table_name>(<column_name>)
```

S08.3.5 Write an SQL script to query or modify data (DML) which are stored in (at most two) database tables

- Queries including: SELECT...FROM, WHERE, ORDER BY, GROUP BY, INNER JOIN, SUM, COUNT, AVG
- Data maintenance including: INSERT INTO, DELETE FROM, UPDATE

SELECT...FROM: Queries fields of a table

```
/* Query several columns of a table */
SELECT <column_name1>, <column_name2>, ...
FROM <table_name>;

/* Query all columns of a table */
SELECT * FROM <table_name>;
```

WHERE: Queries the records fit the condition

```
/* Query several columns of the records fit the condition */
SELECT <column_name1>, <column_name2>, ...
FROM <table_name>
WHERE <condition>;
/* Queries the records fit the condition */
SELECT * FROM WHERE ;
```

LIKE: Search for the specified pattern in the WHERE condition

```
/* SELECT all the columns of the records fit the condition include
the pattern */
SELECT <column_name1>, <column_name2>, ...
FROM <table_name>
WHERE <condition> LIKE <pattern>;

/* SELECT all the columns of the records fit the condition not
include the pattern*/
SELECT <column_name1>, <column_name2>, ...
FROM <table_name>
WHERE <condition> NOT LIKE <pattern>;
```

"%" could be used as a wildcard character in the pattern.

- '%A': the label end with character 'A'
- 'B%': the label start with character 'B'
- '%AB%': the label includes character 'AB'

INSERT INTO: Insert new rows into the table

```
INSERT INTO <table_name>
VALUES(<value1>,<value2>,<value3>,...);

INSERT INTO <table_name>(<column_name1>, <column_name2>, ...)
VALUES(<value1>,<value2>,<value3>,...);
```

DELETE FROM: Remove rows from a table

```
DELDTE FROM <table_name>
WHERE <condition>;
```

UPDATE: Edit rows in a table

```
UPDATE <table_name>
SET <column_name1> = <value1>, <column_nam> = <value2>, ...
WHERE <condition>;
```

INNER JOIN:

```
SELECT <column_name1>, <column_name2>, ...
FROM <table_name1>
INNER Join <table_name2>
ON <table_name1>.<column_name> = <table_name2>.<column_name>;
```

SUM():

```
SELECT SUM(<column_name>) FROM <table_name>;
```

COUNT():

```
SELECT COUNT(<column_name>) FROM <table_name>;
SELECT COUNT(*) FROM <table_name>;
```

AVG():

```
SELECT AVG(<column_name>) FROM <table_name>;
```

ORDER BY:

```
SELECT <column_name1>, <column_name2>, ...
FROM <table_name>
ORDER BY <column_name> ASC|DESC;
```

GROUP BY:

```
SELECT <column_name>, <aggregate_function>(<column_name>)
FROM <table_name>
WHERE <condition>
GROUP BY <column_name>;
```