Structured Query Language (SQL)
INSERT

INSERT INTO table_name [ (column_list) ]
VALUES (data_value_list)

- `column_list` is optional.
- If omitted, SQL assumes a list of all columns in their original CREATE TABLE order.
INSERT

- `data_value_list` must match `column_list` as follows:
  - Number of items in each list must be the same.
  - Must be direct correspondence in position of items in two lists.
  - Data type of each item in `data_value_list` must be compatible with data type of corresponding column.
Example INSERT ... VALUES

Insert a new record into Staff table supplying data for all columns.

```
INSERT INTO staff
VALUES ('SG16', 'Alan', 'Brown',
        '67 Endrick Rd, Glasgow G32 8QX',
        '0141-211-3001', 'Assistant', 'M', '25-May-57',
        8300, 'WN848391H', 'B3');
```
Example INSERT using Defaults

Insert a new record into Staff table supplying data for all mandatory columns.

```
INSERT INTO staff (sno, fname, lname, position, salary, bno)
VALUES ('SG44', 'Anne', 'Jones', 'Assistant', 8100, 'B3');
```
Example INSERT using Defaults

Or

```
INSERT INTO staff
VALUES ('SG44', 'Anne', 'Jones', NULL, NULL,
    'Assistant', NULL, NULL, 8100, NULL, 'B3');
```
UPDATE

UPDATE table_name
SET column_name1 = data_value1
  [, column_name2 = data_value2...]
[WHERE search_condition]

- *table_name* can be name of a base table or an updatable view.
- SET clause specifies names of one or more columns that are to be updated.
**UPDATE**

- **WHERE clause is optional:**
  - If omitted, named columns are updated for all rows in table.
  - If specified, only those rows that satisfy `search_condition` are updated.

- New *data_value(s)* must be compatible with data type for corresponding column.
Example UPDATE All Rows

Give all staff a 3% pay increase.

```
UPDATE staff
SET salary = salary*1.03;
```
Example UPDATE Specific Rows

Give all Managers a 5% pay increase.

```
UPDATE staff
SET salary = salary*1.05
WHERE position = 'Manager';
```

- WHERE clause finds rows that contain data for Managers. Update is applied only to these particular rows.
Example UPDATE Multiple Columns

Promote David Ford (sno = 'SG14') to Manager and change his salary to 18,000.

```
UPDATE staff
SET position = 'Manager', salary = 18000
WHERE sno = 'SG14';
```
DELETE

DELETE FROM table_name
[WHERE search_condition]

- `table_name` can be name of a base table or an updatable view.
- `search_condition` is optional; if omitted, all rows are deleted from table. This does not delete table. If `search_condition` is specified, only those rows that satisfy condition are deleted.
Example DELETE Specific Rows

Delete all viewings that relate to property PG4.

DELETE FROM viewing
WHERE pno = 'PG4';
Example DELETE All Rows

Delete all records from the Viewing table.

DELETE FROM viewing;
CREATE TABLE (Basic)

CREATE TABLE table_name
(col_name data_type [NULL | NOT NULL] [,...])

- Creates a table with one or more columns of the specified `data_type`.
- `NULL` (default) indicates whether column can contain `nulls`.
- With `NOT NULL`, system rejects any attempt to insert a null in the column.
CREATE TABLE (Basic)

- Primary keys should always be specified as NOT NULL.
- Foreign keys are often (but not always) candidates for NOT NULL.
Example CREATE TABLE

CREATE TABLE staff(
sno VARCHAR(5) NOT NULL,
fname VARCHAR(15) NOT NULL,
lname VARCHAR(15) NOT NULL,
address VARCHAR(50),
tel_no VARCHAR(13),
position VARCHAR(10) NOT NULL,
sex CHAR,
dob DATETIME,
salary DECIMAL(7,2) NOT NULL,
nin CHAR(9),
bno VARCHAR(3) NOT NULL);

Example CREATE TABLE

CREATE TABLE property_for_rent(
  pno VARCHAR(5) NOT NULL,
  street VARCHAR(25) NOT NULL,
  area VARCHAR(15),
  city VARCHAR(15) NOT NULL,
  pcode VARCHAR(8),
  type CHAR(1) NOT NULL,
  rooms SMALLINT NOT NULL,
  rent DECIMAL(6,2) NOT NULL,
  ono VARCHAR(5) NOT NULL,
  sno VARCHAR(5),
  bno VARCHAR(3) NOT NULL);

DROP TABLE

DROP TABLE tbl_name [RESTRICT | CASCADE]

e.g. DROP TABLE property_for_rent;

- Removes named table and all rows within it.
- With RESTRICT, if any other objects depend for their existence on continued existence of this table, SQL does not allow request.
- With CASCADE, SQL drops all dependent objects (and objects dependent on these objects).