Database Fundamentals & Design

Structured Query Language (SQL)

- A standard language used to create, maintain and control database.
- It's devided into:
  - ✓ Data Definition Language (DDL).
  - ✓ Data Manipulation Language (DML).
  - ✓ Data Control Language (DCL).

Data types

- A data type determines the type of data that can be stored in a database column. The most commonly used data types are:

✓ Alphanumeric: data types used to store characters, numbers, special characters, or nearly any combination.

✓ Numeric.

✓ Date and Time .

Database Constraints

- Not Null.
- Primary Key.
- Unique Key.
- Referential Integrity (FK).
- Check .

Data Manipulation Language

- Insert.
- Update.
- Delete.
- Select.

INSERT Command

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo

✓ INSERT INTO "table_name" VALUES ("value1", "value2", ...)

- **Insert a New Row:**

INSERT INTO Person VALUES ('Saleh', 'Ahmed', 'Moharam bak', 'Alex.')

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo
Saleh	Ahmed	Moharam bak.	Alex.

INSERT Command (cont.)

- Insert Data in Specified Columns:

Person table

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo

- **Insert a New Row:**

INSERT INTO Person (LastName, City) VALUES ('Hassan', 'Assuit')

Person table

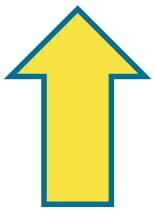
LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo
Hassan			Assuit.

Update Command

- UPDATE "table_name"
  SET "column_1" = {new value}
  [WHERE {condition}]

Example (1)

```
UPDATE Person
SET City= 'Assiut'
```



All records will be updated

Example (2)

```
UPDATE Person
SET City= 'Assiut'

Where FirstName = 'Ahmed'
```

Only records with first name 'Ahmed' will be updated

Update Command (cont.)

✓ Update several Columns in a Row:

LastName	FirstName	Address	City
El-Sayed	Mohamed	Nasr City	Cairo
Saleh	Ahmed	Moharam bak.	Alex.

```
UPDATE Person
SET    Address = '241 El-haram ',  City = 'Giza'
WHERE  LastName = 'El-Sayed'
```

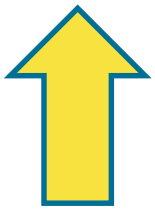
LastName	FirstName	Address	City
El-Sayed	Mohamed	241 El-haram	Giza
Saleh	Ahmed	Moharam bak.	Alex.

Delete Command

- DELETE FROM "table_name"
  [WHERE {condition}]

Example (1)

DELETE FROM Person



All records will be deleted

Example (2)

DELETE FROM Person
Where FirstName = ‘Ahmed’

Only records with first name ‘Ahmed’ will be deleted

Simple Queries

Select <attribute list >
From < table list>
[Where <condition>]

- ✓ select *
  from department;

- ✓ select emp_id, emp_name, dept_id
  from employee;

- ✓ select distinct dept_id
  from employee;

Simple Queries (cont.)

```
Select  dept_id, dept_name
from    department
where   location = ‘Cairo’;
```

Comparison Conditions

- = Equal.
- > greater than.
- >= greater than or equal.
- < less than.
- <= less than or equal.
- <>not equal.

```
Select   last_name, salary
from     employee
where    salary >1000
```

Logical Conditions

- AND.

```
Select   last_name, salary
from     employee
where  city = ‘Assiut’ and salary > 1000;
```

- OR.

```
Select   last_name, salary
from     employee
where  city = ‘Assiut’ OR salary > 1000;
```

- NOT.

```
Select   emp_id, last_name, salary, manager_id
From     employee
where    manager_id NOT IN (100, 101, 200);
```

Other Comparison Conditions

- Between …… AND ….. (between two values - Inclusive).

```
Select   last_name, salary
from     employee
where  salary between 1000 and 3000;
```

- IN (set) (Match any of a list of values)

```
Select    emp_id, last_name, salary, manager_id
From      employee
where     manager_id IN (100, 101, 200);
```

- Like (Match a character Pattern)

```
Select   first_name
from      employee
where   first_name Like '_s%';
```

Arithmetic Expressions

```
Select   last_name, salary, salary + 300
from     employee;
```

- Order of precedence: * , / , +, -
- You can enforce priority by adding parentheses.

```
Select    last_name, salary, 10 * (salary + 300)
from      employee;
```

Order by Clause

- It is used to sort results either in ascending or descending order.

✓
```
Select      fname, dept_id, hire_date
From        employee
Order by    hire_date  [ ASC ];
```

✓
```
Select      fname, dept_id, hire_date
From        employee
Order by    hire_date  DESC;
```

✓
```
Select      fname, dept_id, salary
From        employee
Order by    dept_id, Salary  DESC;
```

Aggregate Functions

COUNT , SUM , MAX, MIN, AVG

```
Select   count (*)
From     employee
```

```
Select  count (Address)
From    employee
```

```
Select  Sum (salary) , Max (salary), Min (salary), Avg (salary)
From    employee
```

Grouping

- Apply aggregate functions to a subgroups of records.

✓ For each department retrieve the department number , the number of employees in the department, and their average salary.

```
Select      dno , count(*) , avg(salary)
From        employee
Group by    dno
```

- Note: Every column in the select clause – which doesn't appear in any aggregate function – must appear in the group by clause.

Grouping (cont.)

- Having clause:

It is used to apply conditions on the subgroups of records.

```
Select      dno , count(*) , avg(salary)
From        employee
Group by    dno
Having      dno > 10
```

Thank You…