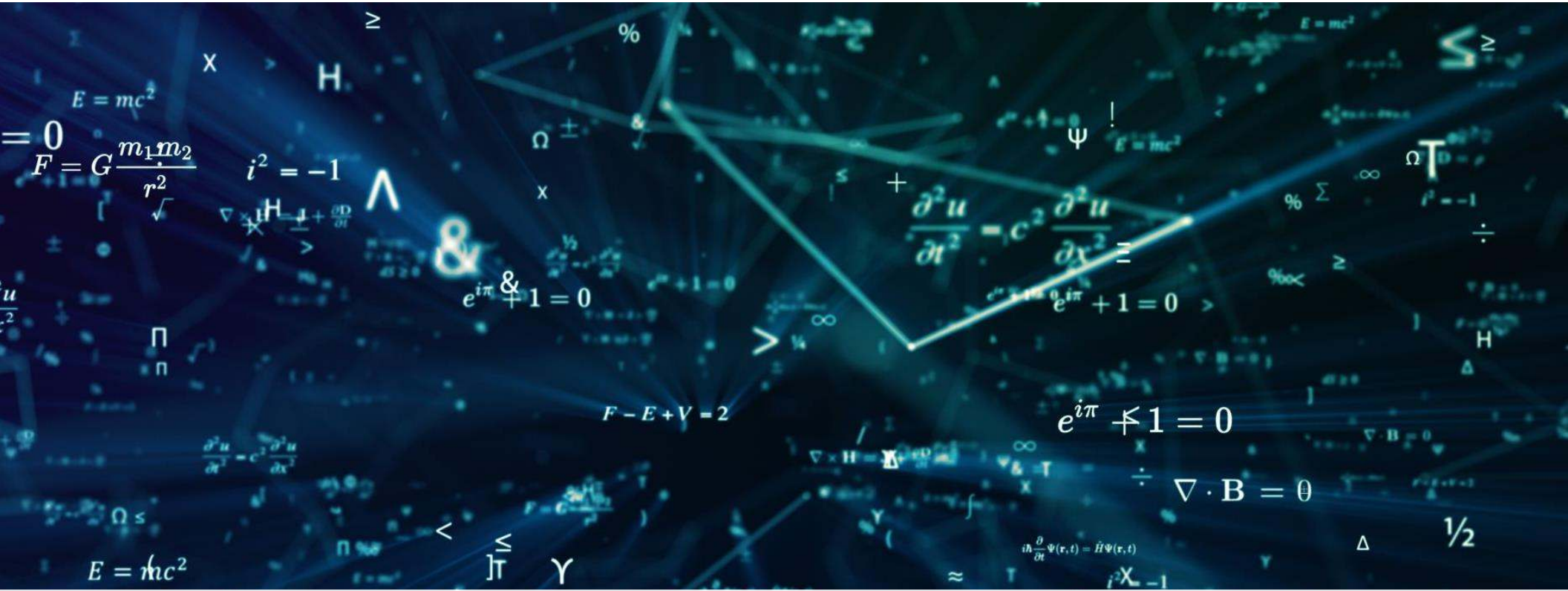


06 BASIC SQL

CS 340: INTRODUCTION TO DATABASE SYSTEMS

Adapted from: Dr. Omar Alomeir 2023

Course instructor:
Dr. Maram Alajlan.

LEARNING GOALS

CLO1: Create a relational database schema in SQL that incorporates key, entity integrity, and referential integrity constraints and uses a declarative query language (SQL) to elicit information from a database. (Skills)

Chapter objectives:

- Write basic SQL queries. Includes: selection, projection, join, ordering, and set operations.
- Define relational algebra and be able to state why it is important

BEFORE WE START..

Make sure you have access to a machine with a functional DBMS, I recommend MySQL. You can use the lab machine, or bring your own laptop to class.

Alternatively, you can use Oracle Apex. We will try to make it work.

We will do many practical exercises throughout the lectures.

We will use Java and JDBC to execute SQL queries, later on.

From now on:

- Tutorials ❎
- Labs ✅

DATABASES SO FAR..

When last we left databases…

We had decided they were great things

We knew how to conceptually model them in ER diagrams

We knew how to logically model them in the relational model

We could formally specify queries Now: how do most people write queries? SQL!

THE SQL QUERY LANGUAGE

We need a standard since relational queries are used by many vendors.

Consists of several parts:

Data Definition Language (DDL) (we have seen this before).

Data Manipulation Language (DML):

- Data retrieval (This is the focus, starts with SELECT).
- Data modification (INSERT, UPDATE, DELETE).

RECALL DDL

name	Ssn	Super_ssn
Asma	10	12
Nada	11	12
Sarah	12	NULL

```
CREATE TABLE EMPLOYEE (
name            VARCHAR(15) ,
Ssn             CHAR(2),
Super_ssn       CHAR(2),
PRIMARY KEY (Ssn),
FOREIGN KEY (Super_ssn) REFERENCES EMPLOYEE(Ssn)
 ON DELETE SET NULL              ON UPDATE CASCADE );
```

THE SQL QUERY LANGUAGE

SQL is declarative, not procedural. What's the difference?

The basic form for a data retrieval query:

```
SELECT <attribute list>
FROM <table list>
WHERE <condition>;
```

Select Name
From Student
where GPA > 3.5;

↑ optional

Student

Sid	Name	DoB	Major
123	Ali	2010	SE
124	Saleh	2010	CS
125	Ahmd	2010	SE
126	Fahad	2013	CS
127	Omar	2012	CS
128	Mhmd	2015	IS
129	Khalid	2015	IS

Course

cid	Text	Description
CS102	Java	OO programming
CS340	DB	database
SE371	Web	Web development

Takes

cid	sid	Grade
CS102	129	90
CS102	128	78
CS102	127	88
CS340	123	85
CS340	124	89
SE371	123	79
SE371	125	91

Example queries:

Q1: SELECT * FROM STUDENT WHERE DoB = 2010;

Q2: SELECT cid, description FROM Course;

*Q3: SELECT * FROM student s, takes t WHERE s.sid = t.sid;*

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Write a query to retrieve the birth date and address of the employee(s) whose name is ‘John B. Smith’.

```
SELECT Bdate, Address

FROM EMPLOYEE

WHERE Fname = ‘John’ AND Minit = ‘B’ AND Lname = ‘Smith’;
```

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Sailors

sid	sname	Rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	12rusty	10	35.0

Write a query to find the names of sailors 'Who have reserved boat number 103

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid = R.sid AND R.bid=103
```

```
Select sname from Sailors
where sid = (select sid from Reserves
              where bid = 103)
```

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Sailors

sid	sname	Rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

The first step is to construct the cross-product between the relations (**join operation**)

sid	sname	Rating	age	sid	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Sailors

sid	sname	Rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

The second step is to apply the qualification *S.sid = R.sid* AND *R.bid=103*

This step eliminates all but the last row from the instance shown in the table below.

sid	sname	Rating	age	sid	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Sailors

sid	sname	Rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

The third step is to eliminate unwanted columns;

sid	sname	Rating	age	sid	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

Reserves

sid	bid	day
22	101	10/10/96
58	103	11/12/96

Sailors

sid	sname	Rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

Write a query to find the names of sailors 'Who have reserved boat number 103

```
SELECT S.sname
FROM Sailors S, Reserves R
WHERE S.sid = R.sid AND R.bid=103
```

Answer:

sname
rusty

CREATING TABLES IN SQL(DDL) - REVISITED

A SQL relation is defined using the **create table** command:

create table *r* (A_1 D_1, A_2 D_2, ..., A_n D_n, (integrity-constraint$_1$),...,(integrity-constraint$_k$))

Integrity constraints can be: primary keys, candidate keys, foreign keys

Example:

```
CREATE TABLE Student (sid      CHAR(20),
                      name     CHAR(20),
                      address  CHAR(20),
                      phone    CHAR(8),
                      major    CHAR(4),
                      primary key (sid))
```

RUNNING EXAMPLE IN YOUR DATABASE

Download the DDL code from LMS.

3 tables:

1. Sailors
2. Boats
3. Reserves

We will use this example throughout the lecture.

PRACTICE QUESTION:SQL PROJECTION

SELECT sname, rating FROM sailors;

Which of these tuples are in the results?

1. (dustin, 7)
2. (dustin, 45)
3. (31, lubber)
4. (lubber, 8)

Sailors

sid	sname	Rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

SELECTION IN SQL

SELECTION is in the WHERE clause

You can use:

1. Attribute names of the relation(s) used in the FROM. comparison operators: =, <>, <, >, <=, >=
2. Apply arithmetic operations: rating*2
3. Operations on strings (e.g., "||" for concatenation).
4. Lexicographic order on strings.
5. Pattern matching: s LIKE p
6. Special format for comparing dates and times.
7. AND, OR to combine operations.

SELECTION EXAMPLE

Query examples: What are the results of the following queries?

```
SELECT * FROM STUDENT WHERE DoB > 2012;
```

Sid	Name	DoB	Major
126	Fahad	2013	CS
128	Mhmd	2015	IS
129	Khalid	2015	IS

```
SELECT * FROM STUDENT WHERE Name LIKE "A%";
```

Sid	Name	DoB	Major
123	Ali	2010	SE
125	Ahmd	2010	SE

Student

Sid	Name	DoB	Major
123	Ali	2010	SE
124	Saleh	2010	CS
125	Ahmd	2010	SE
126	Fahad	2013	CS
127	Omar	2012	CS
128	Mhmd	2015	IS
129	Khalid	2015	IS

ACTIVITY: SELECTION

Consider the table Scores

```
SELECT *
FROM Scores
WHERE RunsFor > 5
```

Which tuple is in the result?

A. (Swallows, Carp, 6, 4)
B. (Swallows, Carp, 4)
C. (12)
D. (*)

Team	Opponent	RunsFor	RunsAgainst
Dragons	Tigers	5	3
Carp	Swallows	4	6
Bay Stars	Giants	2	1
Marines	Hawks	5	3
Ham Fighters	Buffaloes	1	6
Lions	Golden Eagles	8	12
Tigers	Dragons	3	5
Swallows	Carp	6	4
Giants	Bay Stars	1	2
Hawks	Marines	3	5
Buffaloes	Ham Fighters	6	1
Golden Eagles	Lions	12	8

ACTIVITY: SELECTION

Consider the table Scores

```
SELECT *
FROM Scores
WHERE RunsFor > 5
```

Which tuple is in the result?

A. (Swallows, Carp, 6, 4)
B. (Swallows, Carp, 4)
C. (12)
D. (*)

Team	Opponent	RunsFor	RunsAgainst
Dragons	Tigers	5	3
Carp	Swallows	4	6
Bay Stars	Giants	2	1
Marines	Hawks	5	3
Ham Fighters	Buffaloes	1	6
Lions	Golden Eagles	8	12
Tigers	Dragons	3	5
Swallows	Carp	6	4
Giants	Bay Stars	1	2
Hawks	Marines	3	5
Buffaloes	Ham Fighters	6	1
Golden Eagles	Lions	12	8

select Runfor from

SELECTION EXAMPLE (DATES)

events

name	date
A	1941-05-25
B	1942-11-15
C	1943-12-26
D	1944-10-25

name	date
A	1941-05-25
B	1942-11-15

```
Select *
From events
Where date < to_date('1-1-1943','DD-MON-YYYY')
```

DUPLICATES, SETS, AND MULTISETS (BAGS)

As we learned before, duplicates are not allowed in the relational model and every tuple is unique.

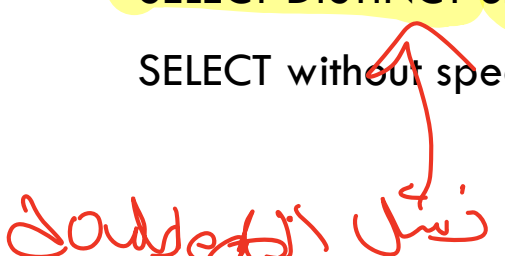
However, SQL usually treats a table not as a set but rather as a multiset; duplicate tuples can appear more than once in a table.

The user may want to see duplicate tuples in the result of a query.

If we want to eliminate duplicate tuples from the result of an SQL query, we use the keyword DISTINCT in the SELECT clause.

SELECT DISTINCT eliminates duplicates, whereas a query with SELECT ALL does not.

SELECT without specifying has implied ALL.



EXAMPLE

```
SELECT sname
FROM sailors s, reserves r
WHERE s.sid = r.sid;
```

SNAME
dustin
dustin
dustin
dustin
lubber
lubber
lubber
rusty

```
SELECT DISTINCT sname
FROM sailors s, reserves r
WHERE s.sid = r.sid;
```

SNAME
rusty
lubber
dustin

RENAMING ATTRIBUTES

If you want the result to have different attribute names, use “AS <new name>” to rename an attribute.

Example: Using Beers(name, manf):

```
SELECT DISTINCT sname AS sailor_name
FROM sailors s, reserves r
WHERE s.sid = r.sid;
```

sailor_name
rusty
lubber
dustin

PATTERN MATCHING

We use the keyword LIKE for pattern matching.

```
SELECT Fname, Lname
FROM EMPLOYEE
WHERE Address LIKE '%Houston,TX%';
```

% replaces an arbitrary number of zero or more characters.

underscore (_) replaces a single character.

Exercise 1: Write a query to retrieve all boats that start with c.

Exercise 2: Write a query to retrieve all sailor IDs who rented a boat that starts with c.

ARITHMETIC OPERATIONS

Show the resulting salaries if every employee is given a 10% raise.

```
SELECT Fname, Lname, 1.1 * E.Salary
FROM EMPLOYEE
;
```

Exercise: Write a query to retrieve all sailor names and ratings, and show their rating out of 5 instead of 10.

BETWEEN OPERATOR

Another comparison operator, used for variable range, is BETWEEN.

Retrieve all employees in department 5 whose salary is between $30,000 and $40,000.

```
SELECT *
FROM EMPLOYEE
WHERE (Salary BETWEEN 30000 AND 40000) AND Dno = 5;
```

Exercise: Write a query to retrieve all sailors who have a rating between 5 and 7.

where (rating Between 5 And 7)

ORDER BY CLAUSE

SQL allows the user to order the tuples in the result of a query by the values of one or more of the attributes that appear in the query result, by using the ORDER BY clause.

Retrieve a list of employees and the projects they are working on, ordered by department and, within each department, ordered alphabetically by last name, then first name.

```
SELECT D.Dname, E.Lname, E.Fname, P.Pname
FROM DEPARTMENT D, EMPLOYEE E, WORKS_ON W, PROJECT P
WHERE D.Dnumber = E.Dno
AND E.Ssn = W.Essn
AND W.Pno = P.Pnumber
ORDER BY D.Dname, E.Lname, E.Fname;
```

ORDER BY CLAUSE (CONTD)

The default order is in ascending order of values. We can specify the keyword DESC if we want to see the result in a descending order of values. The keyword ASC can be used to specify ascending order explicitly.

default

Query from previous slide can be written like this:

```
ORDER BY D.Dname DESC, E.Lname ASC, E.Fname ASC
```

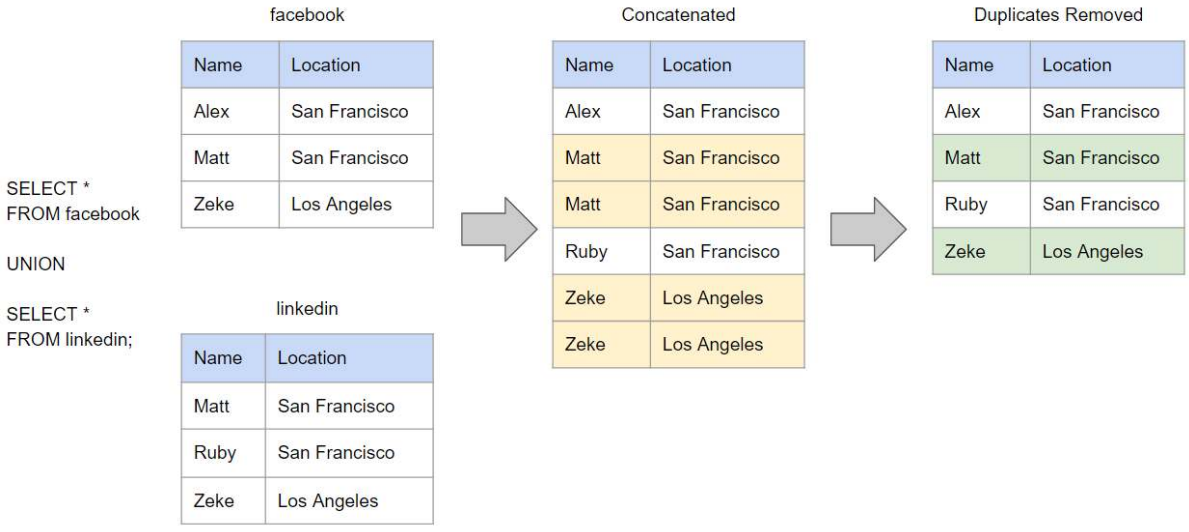
Exercise: Write a query to retrieve sailors in ascending order of age and another query using descending order of rating.

Select *
From Sailors
OrderBy age ← default

Select *
From Sailors

UNION

The UNION operator is a set operator that combines result sets of two or more SELECT statements into a single result set.



UNION ALL

By default, the UNION operator returns the unique rows from both result sets. If you want to retain the duplicate rows, you explicitly use UNION ALL.

```
SELECT *
FROM facebook

UNION ALL

SELECT *
FROM linkedin;
```

facebook

Name	Location
Alex	San Francisco
Matt	San Francisco
Zeke	Los Angeles

linkedin

Name	Location
Matt	San Francisco
Ruby	San Francisco
Zeke	Los Angeles



Result

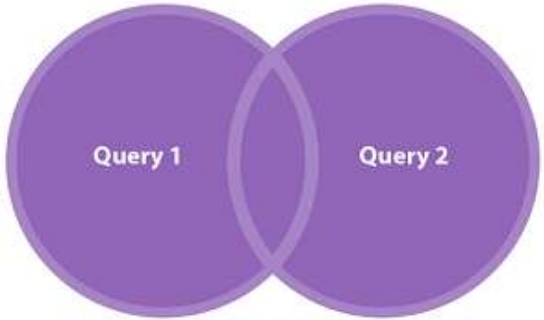
Name	Location
Alex	San Francisco
Matt	San Francisco
Matt	San Francisco
Ruby	San Francisco
Zeke	Los Angeles
Zeke	Los Angeles

MORE SET OPERATORS (EXCEPT/MINUS-INTERSECT)

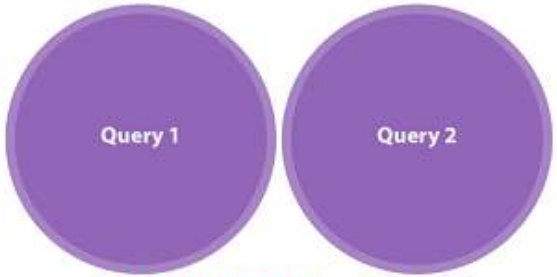
The **EXCEPT/Minus** operator in SQL is used to retrieve all the unique records from the left operand (query), except the records that are present in the result set of the right operand (query).

The **INTERSECT** operator in SQL is used to retrieve the records that are identical/common between the result sets of two or more queries.

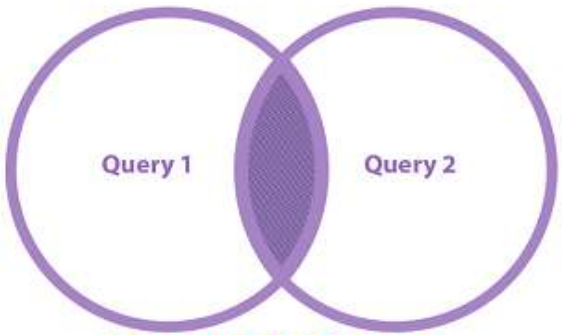
SET OPERATORS



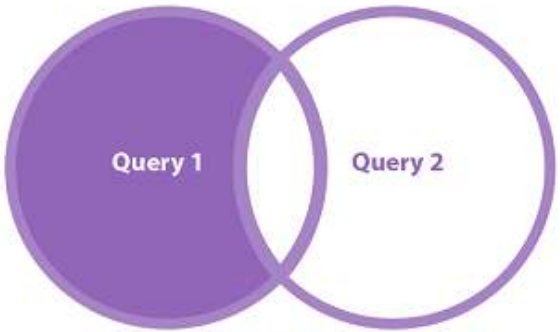
UNION



UNION ALL



INTERSECT



EXCEPT

REVIEW OF BASIC QUERIES

Basic query form:

```
SELECT <attribute list>
FROM <table list>
[ WHERE <condition> ]
[ ORDER BY <attribute list> ];
```

A simple retrieval query in SQL can consist of up to four clauses (that we have seen so far), but only the first two—SELECT and FROM—are mandatory.

INSERT, DELETE, AND UPDATE STATEMENTS IN SQL

In SQL, three commands can be used to modify the database: INSERT, DELETE, and UPDATE.

We discuss each of these in turn.

INSERT

In its simplest form, INSERT is used to add a single tuple (row) to a relation (table).

The values should be listed in the same order in which the attributes were specified in the CREATE TABLE command.

Example:

```
INSERT INTO sailors VALUES(22, 'dustin',7,45.0);
```

Alternatively, we can specify the attributes to insert into.

Example:

```
INSERT INTO sailors(sid, sname, rating) VALUES(22,
'dustin',7);
```

What happens to attribute age here?

INSERT (CONTD)

INSERT commands must meet constraints specified in the CREATE TABLE DDL code.

Try it:

```
INSERT INTO sailors VALUES(22, 'dustin',7,45.0);
INSERT INTO sailors(sname, rating) VALUES('dustin',7);
```

What happens in each case?

Can you think of an INSERT statement that violates referential integrity?

INSERT (CONTD)

A variation of INSERT enables inserting the results of a query into a table.

```
CREATE TABLE WORKS_ON_INFO
( Emp_name VARCHAR(15),
Proj_name VARCHAR(15),
Hours_per_week DECIMAL(3,1) );

INSERT INTO WORKS_ON_INFO ( Emp_name, Proj_name,
Hours_per_week )
SELECT E.Lname, P.Pname, W.Hours
FROM PROJECT P, WORKS_ON W, EMPLOYEE E
WHERE P.Pnumber = W.Pno AND W.Essn = E.Ssn;
```

INSERT (CONTD)

Another variation is to do this with CREATE TABLE:

```
CREATE TABLE D5EMPS LIKE EMPLOYEE
(SELECT E.*
FROM EMPLOYEE AS E
WHERE E.Dno = 5) WITH DATA;
```

All DBMSs have a data loading command from files, so you do not need to manually write all your INSERT statements for every tuple.

Example: `LOAD DATA INFILE` in MySQL

DELETE

The DELETE command removes tuples from a relation.

It includes a WHERE clause to select the tuples to be deleted.

Tuples are explicitly deleted from only one table at a time. However, the deletion may propagate to tuples in other relations if referential triggered actions are specified in the referential integrity constraints of the DDL. Example: ON DELETE CASCADE

Example:

```
DELETE FROM EMPLOYEE
WHERE Lname = 'Brown';
```

Exercise: delete all sailors that are older than 50.

UPDATE

The UPDATE command is used to modify attribute values of one or more selected tuples. It includes a WHERE clause to select the tuples to be deleted.

Tuples are explicitly updated from only one table at a time. However, However, updating a primary key value may propagate to the foreign key values of tuples in other relations. Example: ON UPDATE CASCADE

The SET clause in the UPDATE command specifies the attributes and their new values.

Example:

```
UPDATE EMPLOYEE
SET Salary = Salary * 1.1
WHERE Dno = 5;
```

Exercise: update the age of sailors to increase by 1.

SUMMARY

We have also looked at basic SQL commands.

In the next chapter, we will present the following features of SQL: complex retrieval queries; views; triggers and assertions; and schema modification commands.