INTRODUCTION TO SQL

Engr. Laraib Siddiqui

Introduction

- Structured Query Language is a computer language aimed to store, manipulate and retrieve data stored in relational databases.
- IBM implemented the language, originally called Sequeal (Structured English Query Language), as part of the System R project in the early 1970s.
- It is a declarative query language to create database schemas, insert, update, delete and query information based on a data definition and data manipulation language .
- It further deals with the following issues
  - transaction control
  - integrity constraints
  - authorization
  - views
  - embedded SQL and dynamic SQL

Basic SQL Query Structure

A basic SQL query consists of a SELECT, a FROM and a WHERE clause

- **SELECT**
  - specifies the columns to appear in the result
- **FROM**
  - specifies the relations to be used
- **WHERE**
  - filters the tuples

Select

The SELECT command is the most commonly used command in SQL. It allows database users to retrieve the specific information they desire from an operational database

SQL SELECT Syntax

```
SELECT column_name,column_name
FROM table_name;
```

OR

```
SELECT *

FROM table_name;
```

Example

```
SELECT *
FROM Customer;
```

A '*' can be used in the SELECT clause as a shortcut to get all tuple attributes

```
SELECT name
FROM Customer;
```

Duplicate tuples resulting from a select are not eliminated by default

```
SELECT DISTINCT name
FROM Customer;
```

The DISTINCT keyword can be used to eliminate duplicates

customerID	name	street	postcode	city
1	Max Frisch	Bahnhofstrasse 7	8001	Zurich
2	Eddy Merckx	Pleinlaan 25	1050	Brussels
5	Claude Debussy	12 Rue Louise	75008	Paris
53	Albert Einstein	Bergstrasse 18	8037	Zurich
8	Max Frisch	ETH Zentrum	8092	Zurich

name
Max Frisch
Eddy Merckx
Claude Debussy
Albert Einstein
Max Frisch

name
Max Frisch
Eddy Merckx
Claude Debussy
Albert Einstein

Computed Attributes and Rename

SELECT *name*, *price* * 1.5 AS newPrice
FROM CD;

name	newPrice
Falling into Place	26.85
Carcassonne	23.20
Chromatic	24.75

cdID	name	duration	price	year
1	Falling into Place	2007	17.90	2007
2	Carcassonne	3156	15.50	1993
3	Chromatic	3012	16.50	1993

- Computations can be performed in the SELECT clause
  - multiple numeric attributes can be used in a computation
- The rename operation (AS) is used to rename relations as well as attributes
  - computed columns have no name by default
  - also used when multiple relations have the same attribute names

WHERE

In the WHERE clause we can use five basic predicates (*search conditions*)

- **comparison** - compare two expressions
- **range** - check whether the value is within a specified range of values (BETWEEN)
- **set membership** - check whether the value is equal to a value of a given set (IN)
- **pattern matching** - test whether the expression matches a specifies string pattern (LIKE)
- **check for NULL values** - check whether the expression is a NULL value (IS NULL)

WHERE – Comparison (AND, OR and NOT Operators)

- The AND operator displays a record if all the conditions separated by AND are TRUE.
- The OR operator displays a record if any of the conditions separated by OR is TRUE.
- The NOT operator displays a record if the condition(s) is NOT TRUE.

AND / OR syntax

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 AND/OR condition2 AND/OR condition3 ...;
```

NOT syntax

```
SELECT column1, column2, ...
FROM table_name
WHERE NOT condition;
```

Example

```
SELECT * FROM Customers
WHERE Country='Germany'
AND City='Berlin';
```

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany

```
SELECT * FROM Customers
WHERE City='Berlin' OR
City='München';
```

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
1	Alfreds Futterkiste	Maria Anders	Obere Str. 57	Berlin	12209	Germany
25	Frankenversand	Peter Franken	Berliner Platz 43	München	80805	Germany

```
SELECT * FROM Customers
WHERE NOT
Country='Germany';
```

CustomerID	CustomerName	ContactName	Address	City	PostalCode	Country
2	Ana Trujillo Emparedados y helados	Ana Trujillo	Avda. de la Constitución 2222	México D.F.	05021	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mataderos 2312	México D.F.	05023	Mexico
4	Around the Horn	Thomas Hardy	120 Hanover Sq.	London	WA1 1DP	UK
5	Berglunds snabbköp	Christina Berglund	Berguvsvägen 8	Luleå	S-958 22	Sweden

WHERE – Checking range

```
SELECT name, price
FROM CD
WHERE price BETWEEN 15.0 AND 17.0;
```

cdID	name	duration	price	year
1	Falling into Place	2007	17.90	2007
2	Carcassonne	3156	15.50	1993
3	Chromatic	3012	16.50	1993

name	price
Carcassonne	15.50
Chromatic	16.50

WHERE – Set membership

customerID	name	street	postcode	city
1	Max Frisch	Bahnhofstrasse 7	8001	Zurich
2	Eddy Merckx	Pleinlaan 25	1050	Brussels
5	Claude Debussy	12 Rue Louise	75008	Paris
53	Albert Einstein	Bergstrasse 18	8037	Zurich
8	Max Frisch	ETH Zentrum	8092	Zurich

```
SELECT *
FROM Customer
WHERE city IN ('Zurich', 'Brussels');
```

customerID	name	street	postcode	city
1	Max Frisch	Bahnhofstrasse 7	8001	Zurich
2	Eddy Merckx	Pleinlaan 25	1050	Brussels
53	Albert Einstein	Bergstrasse 18	8037	Zurich
8	Max Frisch	ETH Zentrum	8092	Zurich

```
SELECT *
FROM Customer
WHERE city NOT IN ('Zurich');
```

customerID	name	street	postcode	city
2	Eddy Merckx	Pleinlaan 25	1050	Brussels
5	Claude Debussy	12 Rue Louise	75008	Paris

WHERE - Pattern Matching

The LIKE operator is used for pattern matching

- the *underscore* (_) is a placeholder for a single character
- the *percent* sign (%) is a placeholder for any substring

```
SELECT DISTINCT name
FROM Customer
WHERE name LIKE '%Ein%';
```

name
Albert Einstein

```
SELECT DISTINCT name
FROM Customer
WHERE name LIKE '_l%';
```

name
Claude Debussy
Albert Einstein

WHERE – NULL values

- Missing (unknown) info is represented by NULL values
- The NULL keyword can also be used in predicates to check for null values

```
SELECT *
FROM CD
WHERE price IS NOT NULL;
```

cdID	name	duration	price	year
1	Falling into Place	2007	17.90	2007
2	Carcassonne	3156	15.50	1993
3	Chromatic	3012	16.50	1993

Practice

- Write a SQL statement to display name and commission for all the salesmen.
- Write a SQL query to find the salespeople who lives in the City of 'Paris'. Return salesperson's name, city.
- Write a SQL statement to display all the information of all salesmen.
- Write a SQL query to find the details of those salespeople whose commissions range from 0.10 to 0.12.

Salesman

salesman_id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5007	Paul Adam	Rome	0.13
5003	Lauson Hen	San Jose	0.12

Source: https://www.w3resource.com/sql-exercises

Practice

- Write a SQL query to find the details of the customers who have a grade value above 100. Return customer_id, cust_name, city, grade, and salesman_id.
- Write a query to find all the customers in 'New York' city who have a grade value above 200.
- Write a SQL query to find the customers who belong to either the city 'New York' or not have a grade above 100.
- Write a SQL query to find the details of those customers whose name starts with 'J' and the fourth character is 'l'. Rests may be any character.

Customer

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3007	Brad Davis	New York	200	5001
3005	Graham Zusi	California	200	5002
3008	Julian Green	London	300	5002
3004	Fabian Johnson	Paris	300	5006
3009	Geoff Cameron	Berlin	100	5003
3003	Jozy Altidor	Moscow	200	5007

Source: https://www.w3resource.com/sql-exercises

Practice

- From the given table, write a SQL query to find details of all order excluding combination of ord_date equal to '2012-09-10' and salesman_id higher than 5005 or purch_amt greater than 1000.
- Write a SQL query to display order number, purchase amount, the achieved and unachieved percentage (%) for those order which exceeds the 50% of the target value of 6000.
- From the following table, write a SQL query to find the unique salespeople ID. Return salesman_id.

Orders

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.5	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.5	2012-08-17	3009	5003
70007	948.5	2012-09-10	3005	5002
70005	2400.6	2012-07-27	3007	5001
70008	5760	2012-09-10	3002	5001

Source: https://www.w3resource.com/sql-exercises