



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

200
1822 - 2022

Enhanced Database Design

Man Geni Addysg Uwch yng Nghymru
Where Higher Education Began in Wales

www.uwtsd.ac.uk

Learning Objectives

- Joins
- Stored Procedures
- Indexes
- Triggers
- Implementation in Postgresql



Stored Procedure



- a set of SQL statements stored in the database that can be executed as a single unit.
- can perform operations like inserting, updating, and deleting data.

Stored Procedure



```
CREATE TABLE Books (  
    BookID SERIAL PRIMARY KEY,  
    Title VARCHAR(100),  
    Author VARCHAR(50),  
    CopiesAvailable INT  
);
```

Stored Procedure



```
INSERT INTO Books (Title, Author,  
CopiesAvailable) VALUES  
( 'To Kill a Mockingbird', 'Harper Lee', 5),  
( '1984', 'George Orwell', 8),  
( 'The Great Gats', 'F. Scott Fitzgerald', 3),  
( 'Moby Jack', 'Herman Melville', 7),  
( 'Pride and Prejudice', 'Jane Austen', 6);
```

Stored Procedure

```
CREATE OR REPLACE PROCEDURE
UpdateCopies(book_id INT, new_copies
INT)
LANGUAGE plpgsql
AS $$
BEGIN
    UPDATE Books
    SET CopiesAvailable = new_copies
    WHERE BookID = book_id;
END;
$$;
```

Stored Procedure



```
CALL UpdateCopies(1, 10); -- This  
updates the book with BookID 1 to have  
10 copies.
```

Trigger



A **trigger** is a function that is automatically executed in response to certain events on a particular table, like insert, update, or delete.

Trigger



```
CREATE TABLE Loans (  
    LoanID SERIAL PRIMARY KEY,  
    BookID INT,  
    LoanDate DATE,  
    FOREIGN KEY (BookID)  
REFERENCES Books (BookID)  
);
```

Trigger



```
INSERT INTO Loans (BookID, LoanDate)
VALUES
(1, '2024-10-01'), (2, '2024-10-03'),
(3, '2024-10-05'), (4, '2024-10-07'),
(5, '2024-10-09');
```

Trigger

```
CREATE OR REPLACE FUNCTION
DecreaseCopies()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$ --delimiter that is used to define the block of code
for the procedure.
BEGIN
    UPDATE Books
    SET CopiesAvailable = CopiesAvailable - 1
    WHERE BookID = NEW.BookID;
    RETURN NEW;
END;
$$;
```

Trigger

```
CREATE TRIGGER LoanTrigger  
AFTER INSERT ON Loans  
FOR EACH ROW  
EXECUTE FUNCTION DecreaseCopies();
```

Transaction



A **transaction** is a group of operations that are executed as a single unit. You can either **commit** these operations to the database or **rollback** if something goes wrong.

Transaction - Example



```
CREATE TABLE Members (  
    MemberID SERIAL PRIMARY KEY,  
    FirstName VARCHAR(50),  
    LastName VARCHAR(50),  
    MembershipStatus VARCHAR(20)  
);
```

Transaction - Example

```
BEGIN;  
  
-- Insert a new member  
INSERT INTO Members (FirstName, LastName,  
MembershipStatus)  
VALUES ('Alice', 'Green', 'Active');  
  
-- Update another member's status  
UPDATE Members  
SET MembershipStatus = 'Inactive'  
WHERE MemberID = 1;  
  
-- Commit the transaction to save changes  
COMMIT;
```

Transaction - Example

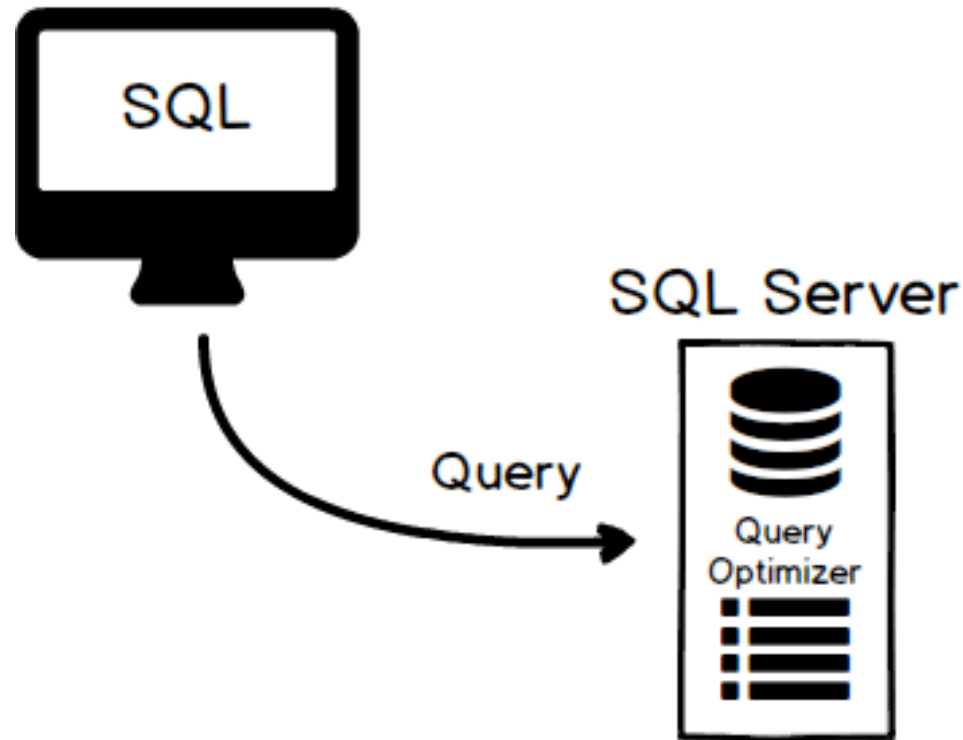


ROLLBACK;

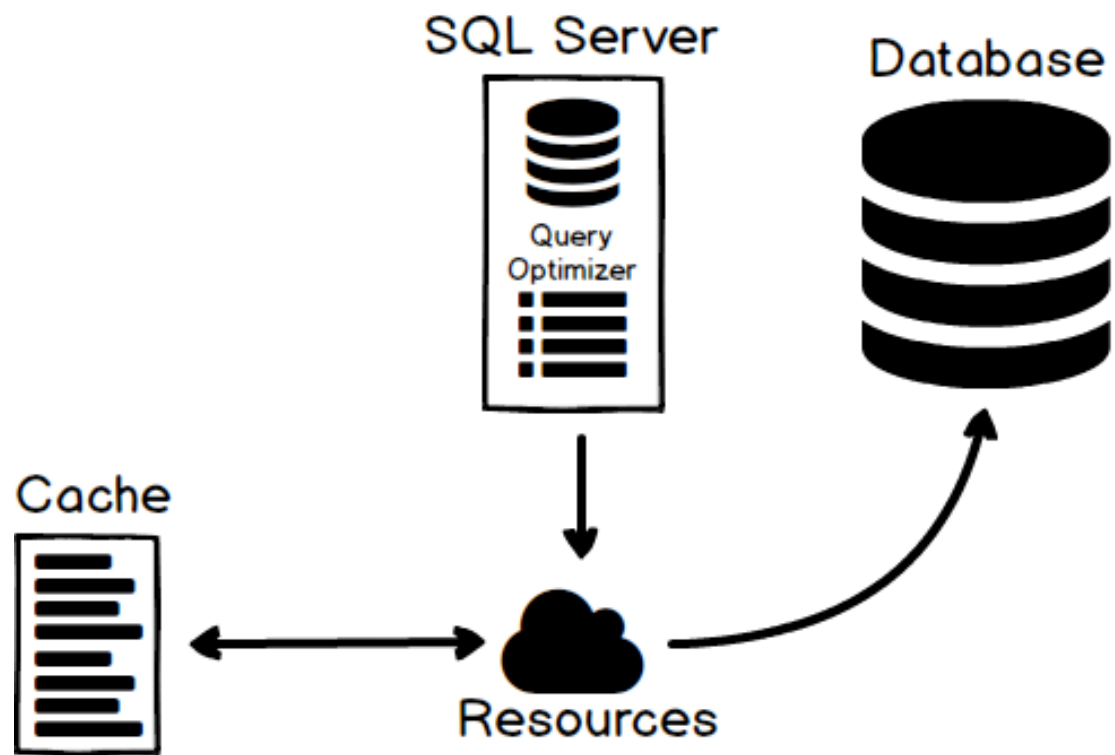
- Enhancing database design
 - Indexing
 - Distributed database-fragmentation
- Transaction management

Enhancing database design

18



- Query Execution plan is a map, most efficient path to data.



```
SELECT * FROM friends WHERE name = 'Zack';
```

friends		
id	name	city
1	Matt	San Francisco
2	Dave	Oakland
3	Andrew	Blacksburg
4	Todd	Chicago
5	Blake	Atlanta
6	Evan	Detroit
7	Nick	New York City
8	Zack	Seattle

Indexing

21

- ❑ create and implement indexes to improve its performance
- ❑ serve as pointers associated to the data of a particular table.
- ❑ A SQL index is used to retrieve data from a database very fast.
- ❑ A SQL index is a quick lookup table for finding records users need to search frequently

Indexing

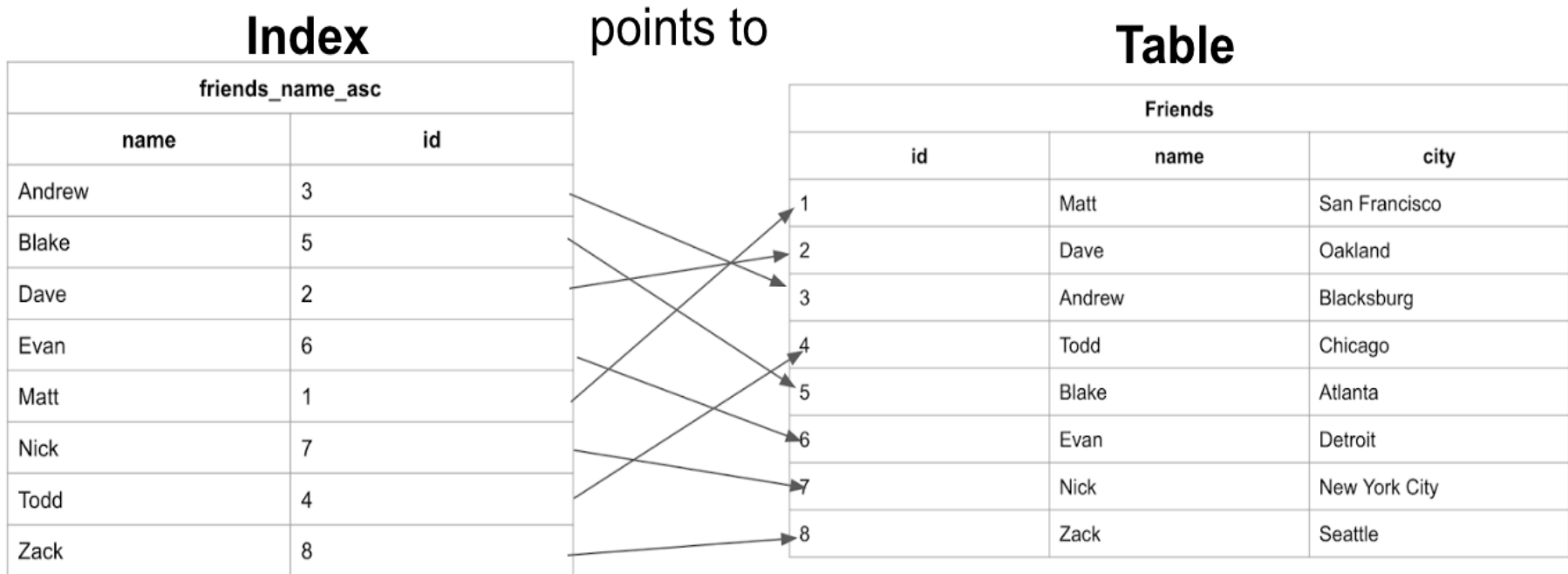
22

- ❑ SQL indexes are primarily a performance tool, so they really apply if a database gets large.
- ❑ storage spaces of an index and the table from which it was created are separate
- ❑ SQL Server supports several types of indexes but one of the most common types is the clustered index. This type of index is automatically created with a primary key

What Exactly is an Index?

23

- The index can be sorted or unsorted and a pointer from each record points to their corresponding record in the original table where the data is actually stored



Indexes



An **index** improves the speed of data retrieval by creating a pointer to the data in a table.

```
CREATE TABLE products (  
    id SERIAL PRIMARY KEY,  
    name VARCHAR(100),  
    category VARCHAR(50),  
    price DECIMAL(10, 2)  
);
```

Indexes



```
INSERT INTO products (name, category, price)
VALUES
('Laptop', 'Electronics', 999.99),
('Smartphone', 'Electronics', 499.99),
('Desk Chair', 'Furniture', 150.00),
('Coffee Table', 'Furniture', 200.00),
('Headphones', 'Electronics', 89.99),
('Monitor', 'Electronics', 299.99),
('Bookshelf', 'Furniture', 120.00);
```

Indexes



create an index on the LastName field in the Members table to speed up searches.

```
CREATE INDEX idx_category ON  
products (category);
```

Analysing Indexes



```
EXPLAIN ANALYZE  
SELECT * FROM products WHERE category = 'Electronics';
```

```
CREATE INDEX idx_category ON products(category);  
EXPLAIN ANALYZE  
SELECT * FROM products WHERE category = 'Electronics';
```

```

1 CREATE TABLE products (
2     id SERIAL PRIMARY KEY,
3     name VARCHAR(100),
4     category VARCHAR(50),
5     price DECIMAL(10, 2)
6 );
7 INSERT INTO products (name, category, price) VALUES
8 ('Laptop', 'Electronics', 999.99),
9 ('Smartphone', 'Electronics', 499.99),
10 ('Desk Chair', 'Furniture', 150.00),
11 ('Coffee Table', 'Furniture', 200.00),
12 ('Headphones', 'Electronics', 89.99),
13 ('Monitor', 'Electronics', 299.99),
14 ('Bookshelf', 'Furniture', 120.00);
15 EXPLAIN ANALYZE
16 SELECT * FROM products WHERE category = 'Electronics';
17 CREATE INDEX idx_category ON products(category);
18 EXPLAIN ANALYZE
19 SELECT * FROM products WHERE category = 'Electronics';
20

```

STDIN

Input for the program (Optional)

```

-----
Seq Scan on products  (cost=0.00..12.62 rows=1 width=356) (actual time=0.032 ms)
  Filter: ((category)::text = 'Electronics'::text)
    Rows Removed by Filter: 3
Planning Time: 0.146 ms
Execution Time: 0.032 ms
(5 rows)

```

CREATE INDEX

QUERY PLAN

```

-----
Seq Scan on products  (cost=0.00..1.09 rows=1 width=356) (actual time=0.025 ms)
  Filter: ((category)::text = 'Electronics'::text)
    Rows Removed by Filter: 3
Planning Time: 0.111 ms
Execution Time: 0.025 ms
(5 rows)

```

Lab Exercise



- ❑ Download Instructions from Moodle



Further Reading :

Chapter 8 Advanced SQL

Database Systems: A Practical Approach to Design, Implementation, and Management Sixth edition by

• Thomas Connolly • Carolyn Begg