**Stored Procedures**

* Stored procedures can have input-output parameters.
* Can return set of rows.
* Use execute.
* Enchances security. Modular programming (can be called many times). Enables delayed binding of objects (can reference an object that doesn’t exist yet). Improves performance.

System stored procedure 🡪 administrative

Sytem extended procedure 🡪 extends functionality

Server level exec 🡪 EXECUTE AS LOGIN

Database level exec 🡪 EXECUTE AS USER

**Using stored procedures:**

CREATE-ALTER-DROP

**Example:**

create procedure sp\_spname

@numberinput int

@charinput nvchar(5)

as

begin

declare @something int

set @something = (select … from … where …=…)

(this is task part)

end

**In the task part:**

* + Use “INSERT INTO tablename” to insert an element
  + Use “DELETE FROM tablename” to delete an item
  + Use “UPDATE tablename” to update a table
  + You can declare variables and use them in the task part.

**Triggers (special stored procedures)**

* There are two types of triggers which are AFTER and INSTEAD OF triggers.
* There are 2 virtual tables: Inserted and Deleted tables.
* Triggers should not return rows of data.
* After triggers: Executed after statement code executes.
* Insted of triggers: Executes instead of the statement code. Like “INSTEAD OF DELETE” or “INSTEAD OF INSERT” OR “INSTEAD OF UPDATE”.
* Multiple triggers may be create for a single event.
* We can include 3 events in one trigger like, AFTER INSERT, UPDATE, DELETE

**Using triggers:**

AFTER INSERT, AFTER DELETE, AFTER UPDATE

create trigger trg\_triggername

on tablename 🡪 on which table action made

after … 🡪 can be insert, update or delete

as

begin

task again as shown in stored procedures

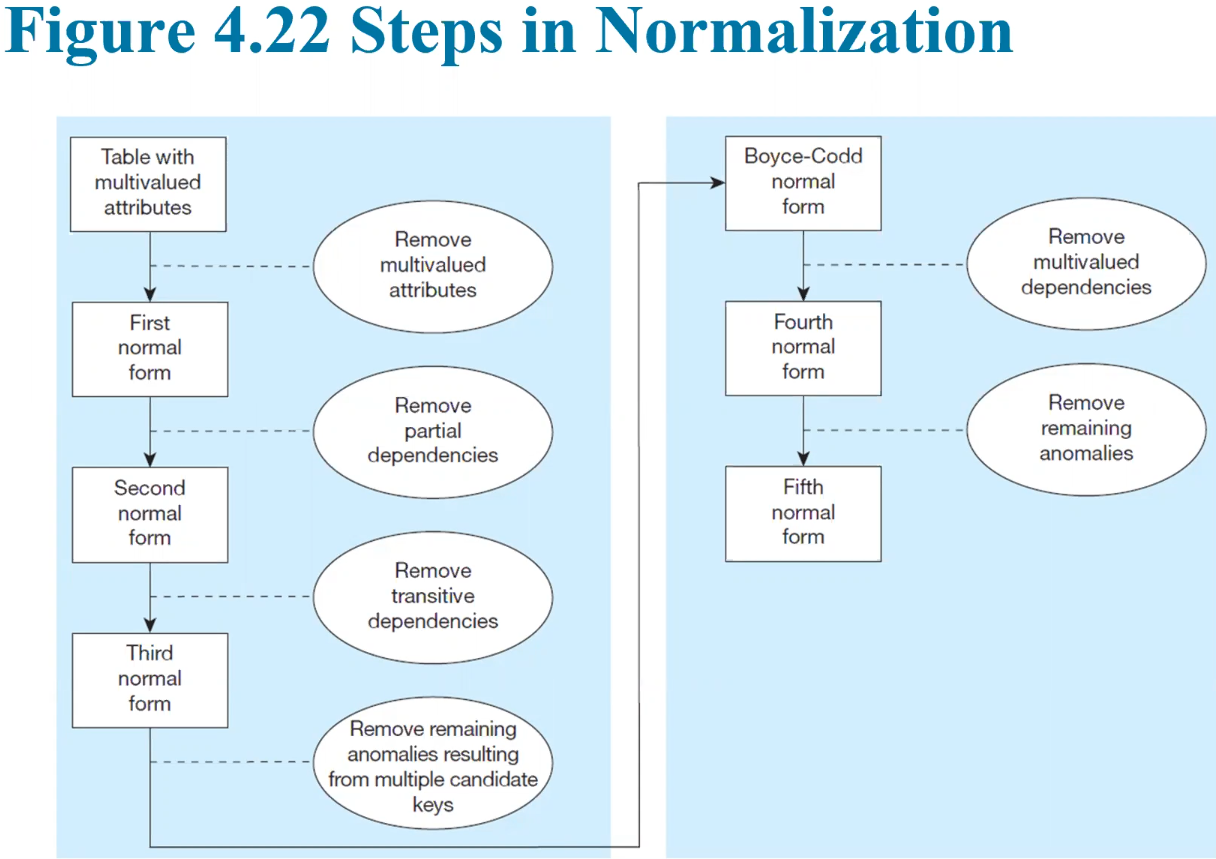
end

**Normalization**

A tool to validate and improve a logical design to avoid unnecessary duplication of data.

Goal is to remove anomalies:

* Insertion Anomaly
* Deletion Anomaly
* Modification Anomaly



**First Normal Form**

* No multivalued attributes
* Every attribute value is atomic
* tablo içeren bir resim

  Açıklama otomatik olarak oluşturuldu

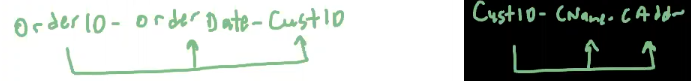
**Second Normal Form**

* 1NF + every non-key attribute is fully dependent on the entire primary key
* tablo içeren bir resim

  Açıklama otomatik olarak oluşturuldu
* OrderID – ProductID – Ordered Quantity will be the table. In OrderID: OrderID – Order Date – Customer ID – Customer Name – Customer Adress. In Product ID: Product Descp., Product Finish, Product SP.

**Third Normal Form**

* 2NF + no transitive dependency as shown below (red)
* metin içeren bir resim

  Açıklama otomatik olarak oluşturuldu
* Make it like that:

**Denormalization**

* Transforming normalized relations into non-normalized
* Can improve performance by reducing of table lookups
* Wasted storage space. Data instegrity/consistency threats
* tablo içeren bir resim

  Açıklama otomatik olarak oluşturuldu
* Decreasing number of joints. Look for key connections.

**Indexing**

* Fast access, retrieval
* Where and group by fields in sql
* Heap (no index)
* Index:
  + Clustered: Logical order, physical order (telephone book)
  + Nonclustered: Logical order (index pages of book)
* create index name on Student(GPA); 🡪 NonClustered index creation (for nonkey attribute)
* create unique clustered index name on Student(StudentID); 🡪 Clustered unique index creation (for primary key)
* You can use table scan where you cant use indexing

**Database Management**

* Information Needed for Physical Design:
  + Normalized relations, including estimates for the range of the number of rows in each table.
  + Data Types, check constraints, defaults.
  + Indexes
  + Response time, data securtiy, recovery, integrity
  + Database management systems used for implementing

**Database Recovery**

* Recovery Facilities:
  + Backup: Copy of entire database
  + Journalizing: Log and images
  + Checkpoint: It is when the system is in a quiet state
  + Recovery: Module that restores the database to a correct condition when a failure occurs. 1.Disk Mirroring 2. Restore 3. Backward Recovery 4. Forward Recovery
* ACID:
  + Atomic: Means transaction must be completed. Means it cannot be further subdivided.
  + Consistend: Any constraints true before, must be the same after
  + Isolated: Changes on database not revealed to users
  + Durable: Changes are permanent
* Responses to Database Failures:
  + Aborted transaction (due abnormal condition)
    - Rollback
    - Rollforward
  + Incorrect data (incorrect but valid data)
    - Rollback
    - Reprocess transaction
    - Compensating transactions
  + System failures (component fails)
    - Disk mirror
    - Rollback
    - Restart from checkpoint
  + Database destruction (totally destroyed)
    - Disk mirror
    - Rollforward
    - Reprocess transaction
* Shared lock: Enable others read, cant update
* Exclusive record: Unable to read
* Deadlock: Stop it before happening
* Deadlock prevention: Lock all records before transaction
* Deadlock resolution: DBMS mechanism that detect and break deadlocks. Matroix of a resoruce
* Transaction: Idenfication of the sequence of steps of a well-defined business activity.
* Hardware failure, loss or corruption of data, maintenance downtime, network-related problems

**SQL Notes**

* SUM(columnname), MAX(columnname), MIN(columnname), AVG(columnname).
* select concat(‘something’ , ‘something’) 🡪 brings them together as 1 string (we can bring them together without using concat). Can also be used like select concat(fname, lname, ‘:’, dname) from student
* select concat\_ws(‘/’, ‘date: 02’, ‘01’, ‘2001’) 🡪 output: date: 02/01/2001
* select left(‘string’, num). Select left(city, 3) from student.
* Select len(‘string’). Select upper(‘string’). Select lower(‘STRING’)
* LTRIM RTRIM yanlardan kesiyor. Select trim(‘string’). Select trim(‘letter’ from ‘string’)
* Select replace(‘string’, ‘letter1’, ‘letter2’)
* Select reverse(‘string’)
* Select substring(‘string’, num1, num2)
* Select isnull(firstone, secondone). Eğer 1 varsa onu bastırır, yoksa 2yi bastırır
* Yeni table oluşturma:
  + Create table employee(

ssn int primary key,

fname char(50),

lname varchar(50)

);

* Select getdate() as date, sysdatetime() as sysdate
* Select Datepart(year, ‘2020-11-25 12:50’) 🡪 output: 2020
* Select Datediff(year, ‘2020-11-25 12:50’, ‘2018-11-25 12:50’) 🡪 2
* Select datalength(variable)
* Select Case (S.gender)

When (‘F’) Then ‘Female’

When (‘M’) Then ‘Male’ End GENGERLONG, S.\*

From STUDENT S

* 
* İif(condition, ‘TRUE’, ‘FALSE’)
* We can create multiple nonclustered index in one table, but we can only create one clustered index in one table. Primary key is an clustered index.