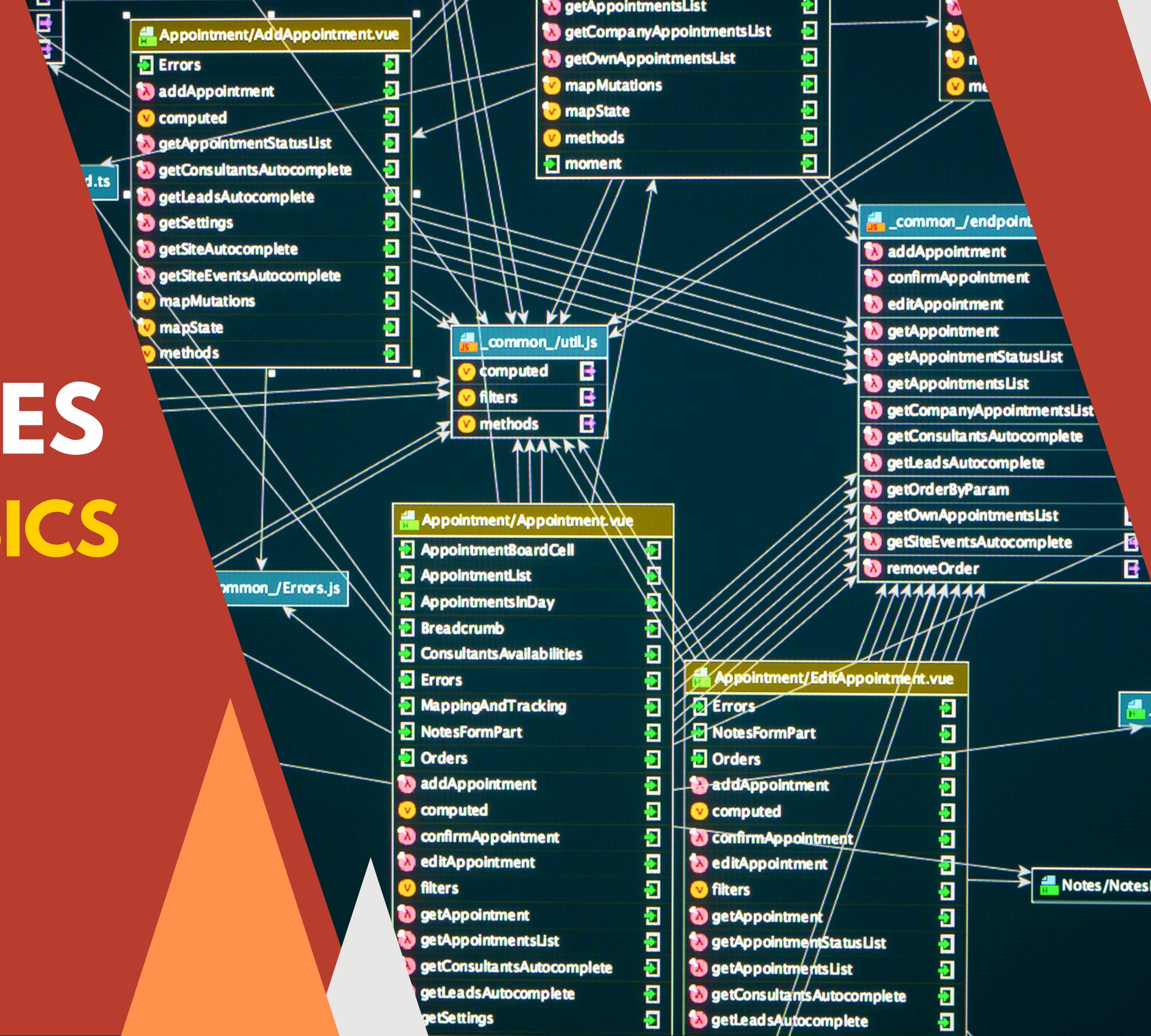




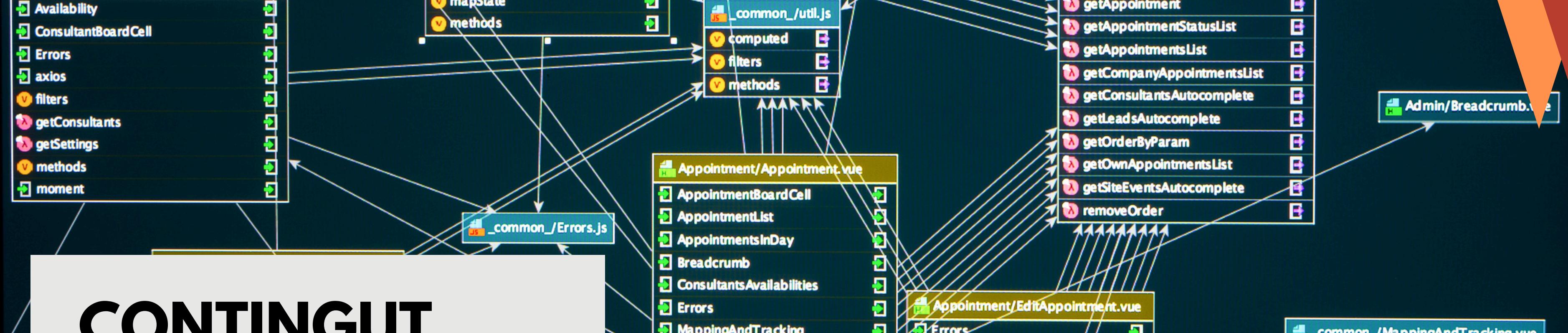
Programació i Tractament
de Dades I

BASE DE DADES CONCEPTEΣ BÀSICS



**“LA INFORMACIÓN ES EL
ACEITE DEL SIGLO XXI, Y LA
ANALÍTICA ES EL MOTOR DE
COMBUSTIÓN.”**

– PETER SONDERGAARD



CONTINGUT

01 QUÈ ÉS UNA BASE DE DADES?

04 LLENGUATGE SQL

02 TIPUS DE BASE DE DADES

05 RESUM

03 MODELATGE DE LES BD

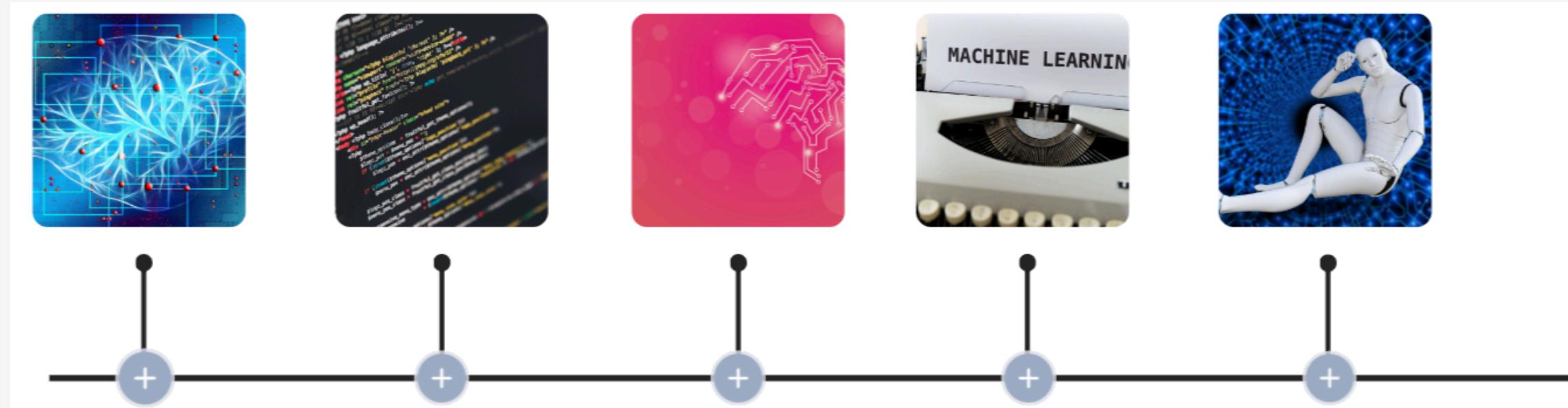
01 QUÈ ÉS UNA BASE DE DADES?

**COL·LECCIÓ ORGANITZADA
D'INFORMACIÓ O DADES**

Les bases de dades permeten emmagatzemar grans quantitats d'informació de manera estructurada, per tal que pugui ser fàcilment recuperades, gestionades i actualitzades. Aquestes dades poden estar relacionades entre elles i sovint es troben estructurades en taules, amb files i columnes.



**PODEU DIR BASES DE DADES
QUE TENIU EN EL VOSTRE DIA A
DIA?**



HISTÒRIA

Els Primers Dies (Anys 1960-1970): Necessitat de gestionar grans quantitats de dades de manera eficient. Els sistemes de fitxers plans eren limitats i inflexibles.

Evolució cap al Model Relacional (Anys 1970-1980): Les dades s'organitzen en taules (relacions) que es poden enllaçar mitjançant claus primàries i estrangeres. Es va desenvolupar el llenguatge SQL, que es va convertir en l'estàndard per gestionar i consultar bases de dades relacionals.

L'era de les Bases de Dades NoSQL (Anys 2000-Present): Amb l'augment d'Internet, el Big Data, i les aplicacions a gran escala, van sorgir limitacions en les bases de dades relacionals. Això va donar lloc al desenvolupament de bases de dades NoSQL, que ofereixen flexibilitat en l'estructura de les dades i són capaços de gestionar grans volums d'informació.

Tendències Recents: Adopció de solucions de bases de dades al núvol i bases de dades especialitzades en la gestió d'esdeveniments i fluxos de dades.

IMPORTÀNCIA

Les bases de dades són essencials en gairebé tots els àmbits de la tecnologia i el negoci. Permeten l'emmagatzematge sistemàtic de dades, la seva recuperació ràpida i la gestió eficient, la qual cosa és crucial en entorns on es manegen grans volums d'informació.

- **Eficiència:** Permeten l'emmagatzematge i accés a dades de manera ràpida i eficient.
- **Organització:** Faciliten l'estructuració de dades, que permet que aquestes siguin fàcilment accessibles i manipulables.
- **Integritat:** Les bases de dades asseguren que les dades es mantinguin correctes i coherents mitjançant restriccions i regles de validació.
- **Seguretat:** Proporcionen mecanismes per protegir les dades contra accessos no autoritzats i pèrdues, mitjançant controls d'accés, encriptació, còpies de seguretat, etc.

02

TIPUS DE BASES DE DADES

CLASSIFICACIÓ SEGONS EL SEU MODEL
DE DADES, ARQUITECTURA O GESTIÓ DE
LA INFORMACIÓ.

DATABASE



BD RELACIONALS

- Les bases de dades relacionals emmagatzemen dades en taules, on cada taula és una col·lecció de registres (files) i cada registre conté camps (columnes).
- Utilitzen el model relacional, que estructura les dades en relacions que es poden manipular i consultar mitjançant el llenguatge SQL (Structured Query Language).
- Els conceptes de clau primària i clau forana són fonamentals en les bases de dades relacionals, que permet enllaçar taules entre elles i assegurar la integritat referencial.
- Exemples: MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server

BD RELACIONALS

AVANTATGES

- Alta consistència i integritat de les dades gràcies a les normes de normalització.
- SQL és un llenguatge estandarditzat i molt estès, amb una sintaxi clara i potent.
- Són ideals per a aplicacions on la integritat de les dades és crítica, com en sistemes financers, ERP, i aplicacions de gestió empresarial.

DESAVANTATGES

- Escalabilitat limitada en comparació amb altres models, especialment quan es tracta de grans volums de dades o aplicacions distribuïdes.
- Pot ser menys flexible per a certs tipus de dades no estructurades o semiestructurades.

BD NOSQL

Les bases de dades NoSQL són un grup divers de sistemes de gestió de bases de dades que no segueixen el model relacional. Estan dissenyades per a la gestió de grans volums de dades, escalabilitat horitzontal, i flexibilitat en la gestió de dades no estructurades o semiestructurades.

Tipus:

- **Bases de Dades de Documents:** Emmagatzemen dades com documents, normalment en formats com JSON, BSON o XML. Cada document pot tenir una estructura diferent, cosa que ofereix una gran flexibilitat.
- **Bases de Dades de Clau-Valor:** Emmagatzemen dades com a parelles clau-valor, on una clau única apunta a un valor que pot ser qualsevol tipus de dades, com un objecte, cadena o llista.

BD NOSQL

AVANTATGES

- Escalabilitat horitzontal que permet afegir més servidors per gestionar més dades.
- Flexibilitat per emmagatzemar i gestionar dades no estructurades o semiestructurades.
- Sovint tenen un rendiment més alt en aplicacions que no requereixen una forta consistència de les dades.

DESAVANTATGES

- Pot faltar-los la consistència i integritat de dades que ofereixen les bases de dades relacionals.
- La manca d'estandardització pot dificultar la migració o integració amb altres sistemes.
- No sempre són la millor opció per a aplicacions amb dades altament relacionades o transaccions complexes.

BD ORIENTADES A OBJECTES

- Aquestes bases de dades emmagatzemen dades en forma d'objectes, tal com es defineixen en la programació orientada a objectes (OOP). Cada objecte pot contenir dades (atributs) i mètodes (funcions associades).
- El model orientat a objectes és útil per a aplicacions que necessiten gestionar dades complexes, com dades multimèdia o models científics.

BD ORIENTADES A OBJECTES

AVANTATGES

- Integració directa amb llenguatges de programació orientats a objectes, com Java, C++, i Python.
- Millor representació de dades complexes, amb suport per a herència, encapsulació i polimorfisme.

DESAVANTATGES

- Menys eficiència en aplicacions on les dades són principalment relacionals.
- Més complexitat en el disseny i implementació en comparació amb les bases de dades relacionals.

BD DISTRIBUÏDES

- Una base de dades distribuïda és aquella en la qual les dades estan distribuïdes a través de múltiples servidors o nodes, que poden estar localitzats en diferents ubicacions geogràfiques.
- Aquestes bases de dades estan dissenyades per oferir escalabilitat, disponibilitat i tolerància a fallades.

BD DISTRIBUÏDES

AVANTATGES

- Alta disponibilitat i tolerància a fallades, ja que les dades es repliquen en diversos nodes.
- Escalabilitat horitzontal, que permet gestionar grans volums de dades afegint més nodes.

DESAVANTATGES

- Major complexitat en la gestió i sincronització de dades entre nodes.
- Problemes de consistència eventual, on les dades poden no estar immediatament sincronitzades entre tots els nodes.

03

MODELATGE DE LES BD

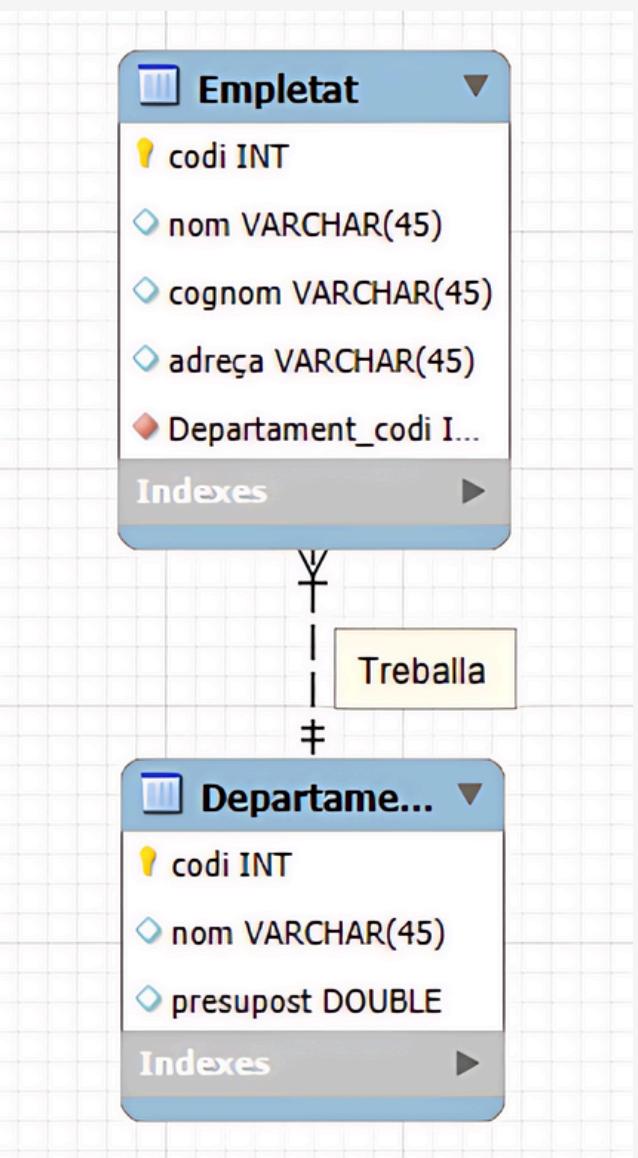
PROCÉS DE DISSENY I ORGANITZACIÓ DE L'ESTRUCTURA DE LES DADES

Un bon modelatge de bases de dades és fonamental per garantir que el sistema sigui eficient, flexible i mantingui la integritat de les dades.



CONCEPTE

- El modelatge de dades implica la creació d'una representació abstracta de les dades que es gestionaran en una base de dades. Aquest procés es duu a terme abans de la implementació de la base de dades i serveix com a guia per a la seva construcció.
- L'objectiu principal del modelatge de dades és definir les estructures de les dades de manera que reflecteixin correctament els requisits del negoci, assegurant que les dades es puguin emmagatzemar, accedir i gestionar de manera eficient.



TIPUS

- **Model Conceptual:** És una representació d'alt nivell que descriu les entitats i relacions generals sense detalls tècnics. El model conceptual és útil per entendre les necessitats del negoci i les relacions entre diferents elements de les dades.
- **Model Lògic:** Detalla l'estructura de la base de dades, incloent-hi taules, columnes, tipus de dades, i relacions entre taules, sense especificar detalls d'implementació específics de la tecnologia.

- **Model Físic:** Descriu com s'implementaran els elements del model lòtic en un sistema de gestió de bases de dades (SGBD) concret, incloent-hi consideracions de rendiment, particionament, índexs, etc.



MODEL ENTITAT-RELACIÓ (ER)

- El Model Entitat-Relació (ER) és una metodologia de modelatge de dades que utilitza diagrames per representar les entitats d'un sistema i les relacions entre elles. És una de les tècniques més utilitzades en el disseny de bases de dades.
- El model ER ajuda a visualitzar la estructura de les dades i les seves interconnexions, la qual cosa facilita la comprensió i comunicació entre desenvolupadors, analistes i usuaris.

ELEMETS

- **Entitats:** Representen objectes o coses del món real que tenen una existència independent i es volen representar dins de la base de dades. Exemples d'entitats inclouen "Client", "Producte", "Ordre", etc.
- **Atributs:** Són propietats o característiques de les entitats. Per exemple, l'entitat "Client" podria tenir atributs com "Nom", "Adreça", "Telèfon", etc.
- **Relacions:** Descriuen com interactuen les entitats entre elles. Per exemple, una relació "Compra" podria existir entre les entitats "Client" i "Producte".
- **Claus:** Són atributs o conjunts d'atributs que identifiquen de manera única una entitat o una relació. Les claus primàries s'utilitzen per identificar de manera única un registre dins d'una taula, mentre que les claus estrangeres s'utilitzen per enllaçar taules entre elles.



NORMALITZACIÓ DE DADES

- La normalització és el procés d'organització de les dades dins d'una base de dades per reduir la redundància i millorar la integritat de les dades. Es duu a terme dividint les taules grans en taules més petites i enllaçant-les mitjançant relacions.
- Aquest procés es basa en diverses formes normals, cadascuna amb un conjunt de regles que s'han de seguir per garantir un disseny òptim de la base de dades.



FORMES NORMALS

- **Primera Forma Normal (1NF):**

Assegura que cada columna conté només valors atòmics, és a dir, que cada cel·la de la taula conté només un valor indivisible.

- **Segona Forma Normal (2NF):**

Estableix que la base de dades ha de complir la 1NF i, a més, tots els atributs no clau han de dependre completament de la clau primària.

- **Tercera Forma Normal (3NF):**

Requereix que la base de dades compleixi la 2NF i que tots els atributs no clau siguin independents entre si i només depenguin de la clau primària.



NORMALITZACIÓ

AVANTATGES

- **Reducció de la Redundància:** Minimitza la duplicació de dades, la qual cosa redueix l'espai d'emmagatzematge i evita inconsistències.
- **Millora de la Consistència:** Redueix les anomalies en les actualitzacions, eliminacions i insercions, assegurant que les dades romanguin coherents.
- **Facilitació del Manteniment:** Una base de dades ben normalitzada és més fàcil de gestionar i mantenir, ja que les dades estan organitzades de manera lògica i estructurada.

DESAVANTATGES

- **Complexitat en les Consultes:** A mesura que es normalitza més una base de dades, poden aparèixer més taules, la qual cosa pot complicar les consultes SQL, especialment quan es requereixen moltes unions (joins).
- **Rendiment:** En alguns casos, la normalització pot afectar el rendiment de les consultes, ja que pot ser necessari accedir a múltiples taules per obtenir la informació desitjada.



CLAUS PRIMÀRIES I FORANES

Les claus primàries i foranes s'utilitzen per mantenir la integritat referencial en una base de dades, assegurant que les relacions entre taules siguin coherents.

Permeten realitzar consultes complexes que involucren múltiples taules, enllaçant dades de manera efectiva.



CLAUS

PRIMÀRIES

- **Una clau primària** és un atribut o un conjunt d'atributs que identifica de manera única un registre dins d'una taula. Cada fila d'una taula ha de tenir un valor únic per a la clau primària, i aquest valor no pot ser nul.
- **Exemple:** En una taula "Estudiants", l'atribut "ID d'Estudiant" podria ser la clau primària, ja que cada estudiant té un identificador únic.

FORANES

- **Una clau forana** és un atribut o un conjunt d'atributs que enllaça una taula amb una altra taula. La clau forana conté valors que corresponen als valors de la clau primària en la taula relacionada.
- **Exemple:** Si una taula "Reserves" té un camp "ID de Client" que es refereix a l'"ID de Client" de la taula "Clients", aquest camp seria una clau forana.



04

LLENGUATGE SQL

**LLENGUATGE ESTÀNDAR UTILITZAT
PER INTERACTUAR AMB
BASES DE DADES RELACIONALS**

Permet als usuaris definir l'estructura de les bases de dades, manipular les dades emmagatzemades i controlar l'accés a aquestes dades.



COMPONENTS PRINCIPALS

DDL (Data Definition Language): Aquest component d'SQL s'utilitza per definir i gestionar l'estructura de les bases de dades i les seves taules.

DML (Data Manipulation Language): El DML s'utilitza per manipular les dades dins de les taules. Les operacions més comunes inclouen la inserció, la consulta, l'actualització i l'eliminació de dades.

DCL (Data Control Language): El DCL s'utilitza per controlar l'accés a les dades dins de la base de dades, gestionant permisos i privilegis dels usuaris.

TCL (Transaction Control Language): El TCL s'encarrega de gestionar les transaccions dins d'una base de dades, assegurant que les operacions es duguin a terme de manera segura i consistent.



ESTRUCTURA

La sintaxi d'una consulta SQL segueix una estructura clara i lògica. La comanda més comuna és SELECT, que s'utilitza per recuperar dades d'una o més taules.

Aquesta consulta selecciona columnal i columna2 de la taula nom_de_taula, aplicant una condició especificada i ordenant els resultats segons columnal.

```
SELECT columna1, columna2  
FROM nom_de_taula  
WHERE condició  
ORDER BY columna1 ASC;
```





CLÀUSULES MÉS COMUNS

- **SELECT:** Selecciona els registres a mostrar.
- **FROM:** Indica les taules on estan els registres
- **WHERE:** Filtra els registres que compleixen una condició específica.
- **ORDER BY:** Ordena els resultats segons una o més columnes.

```
SELECT columna1, columna2  
FROM nom_de_taula  
WHERE condició  
ORDER BY columna1 ASC;
```

FUNCIONS SQL

- **COUNT:** Retorna el nombre de registres.
- **SUM:** Suma els valors numèrics d'una columna.
- **AVG:** Calcula la mitjana dels valors d'una columna
- **MAX:** Retorna el valor màxim.
- **MIN:** Retorna el valor mínim.

```
SELECT COUNT(*) FROM clients;
```

```
SELECT SUM(Preu) FROM Reserves;
```

```
SELECT AVG(Preu) FROM Reserves;
```

```
SELECT MAX(Preu) FROM Reserves;
```

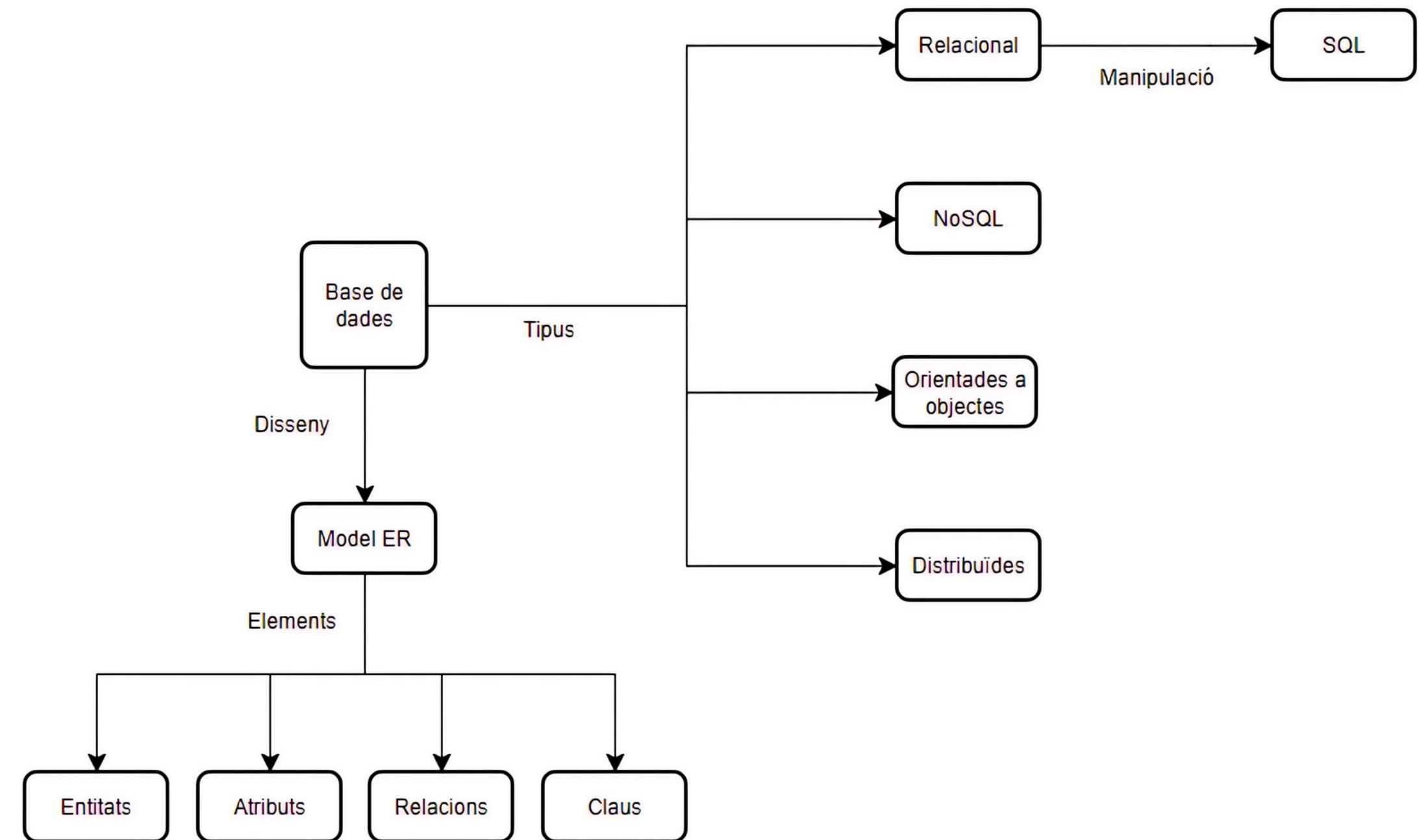
```
SELECT MIN(Preu) FROM Reserves;
```



05

RESUM

IMPORTANT



**PRACTICE
MAKES
PERFECT**

