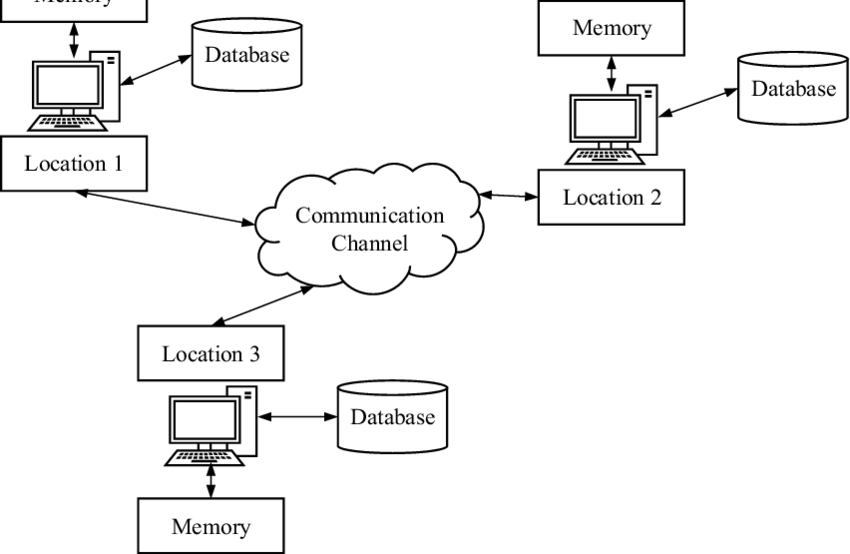
**9. Architektura databázových systémů, správa, služby, systémový katalog**

**MySQL architektura**

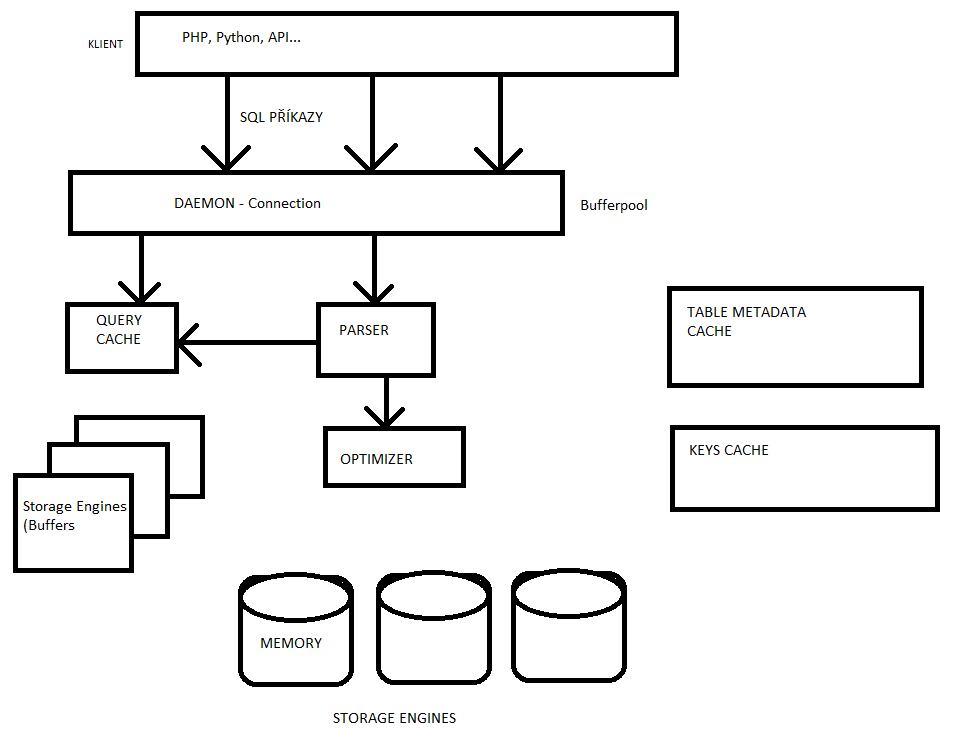
* **Flexibilní pro různá úložiště** (MySQL)
* Chování a zpracování vstupů
* Záleží na druhu aplikace

**Typy architektur**

1. **Fyzická architektura**
   * Ta je založena na:
     1. Adresářích souborů
     2. Data + struktura



1. **Logická architektura**
   * Relační DB v současné době používají architekturu
     1. **KLIENT** -> **SERVER** = **Aplikace**/**Databáze**



Funguje takto:

* + **KLIENT** = Utilita pro připojení k serveru
  + **SERVER** = Instrukce, kde jsou data uložena (databáze)
  + **MySQLD** = Server **DAEMON**, který běží na pozadí
    - 1. **Spravuje příchozí a odchozí požadavky**
      2. **Více vláknový proces**
      3. **Session – pro každé připojení zvlášť**
      4. **Každý klient dostane svoje ID, pod kterým běží požadavky**
  + **Parser**
    1. Kontroluje SQL syntaxe
    2. Analyzuje správnost v SQL příkazu
    3. **Generuje a přidává SQL\_ID každému příkazu**
    4. Kontroluje uživatelské oprávnění
  + **Optimizer**
    1. Vytváří efektivní prováděcí plán SQL požadavků
    2. Organizuje pořadí vyhotovení dotazů a kontroluje uživatelské oprávnění
  + **Metadata CACHE**
    1. Paměť pro metadata
    2. Informace o strukturách objektů a statistiky
  + **Query CACHE**
    1. Paměť, kde jsou uloženy sdílené identické dotazy
       1. Sdílená paměť pro všechny sessions
       2. Dotaz/Příkaz jednoho klienta může být použít i pro jiné
       3. Pokud je stejný dotaz nalezen, Server vezme výsledek z Query CACHE a neprovádí se PARSE dotazu
       4. PARSER pouze dotazu zadá nové **SQL\_ID**
    2. Query CACHE je příklad použití pre-cache dat
  + **Key CACHE**
    1. Paměť indexů tabulek
    2. **Pokud je velikost indexů malá (málo atributů)**
       1. Paměť obsahuje strukturu i data indexu
    3. **Pokud je větší**
       1. Obsahuje pouze strukturu
       2. Data jsou uložena ve storage enginu
    4. MyISAM (ISAM -> Index Sequential Access Method)
       1. Algoritmus vytvořený IBM
       2. Získávání info pomocí indexů
  + **Storage ENGINE**
    1. InnoDB
       1. Transakce podporovaný příkaz k datům
       2. Data storage rozdělený podle tablespace
       3. Row locking

**Systémový katalog**

* Data dictionary
* **Soubor speciálních tabulek v DB**
* **Vytváří, aktualizuje a vlastní** samotný DB systém (Systém, SYS, DBA)
* Tyto tabulky obsahují data, která **popisují databáze**
  + Informace o existujících DB objektech, jejich vlastnících
  + Datumy vytvoření a poslední změny
  + Seznamy tabulek, jejich atributy
  + Datové typy, omezení
  + Práva
  + A další informace související s fyzickým uložením tabulek
* **PŘ:** Zjistěte dodavatele a produkty které dodávají
  + SELECT název,cena FROM Produkt INNER JOIN …
  + **V systémovém katalogu bude nutno ověřit:**
    - Existenci názvů uvedených tabulek či pohledů v DB uživatele, který dotaz zadal
      * Pokud nebude -> Ověřit, zda neexistují synonyma (**aliasy**) ukazující na objekt s jiným jménem
    - Ověření, že uživatel má právo k **SELECT** **těchto** **tabulek**
    - Kontrola, zda uvedené atributy existují v daných tabulkách a zda nejsou stejné názvy
    - Určení datových typů pro požadované atributy
* Aktualizace systémového katalogu zajišťuje databáze na základě uživatelem prováděných akcí na DB objektech
  + **Každá DDL operace provede změnu i v systémovém katalogu**
* Standart ANSI/ISO
  + Nespecifikuje strukturu a obsah syst. Tabulek (v různých systémech se značně liší), ale definuje standard SQL pohledy na tabulky vytvořené katalogem jež obsahují všechny objekty DB, ke kterým má přístup uživatel.
  + Těmto pohledům se říká INFORMAČNÍ SCHÉMA
    - Většina DB systémů má implementováno toto “jednoduché” informační schéma