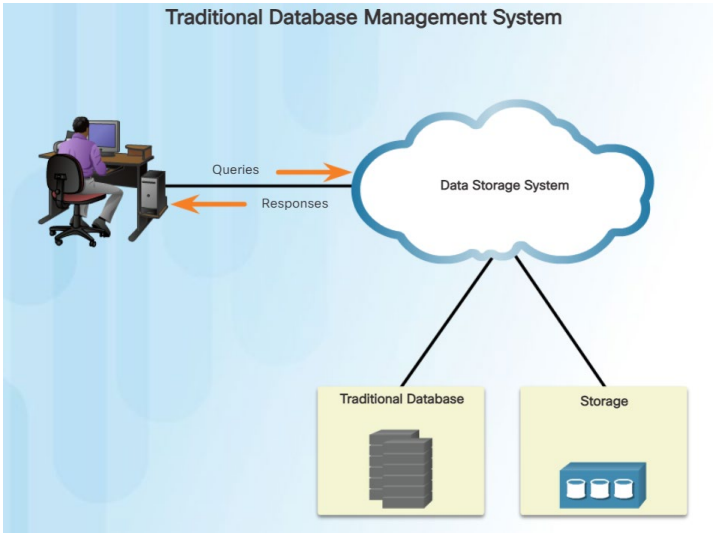
Správa big dat

Infrastruktura Big dat

Tradiční databázový model

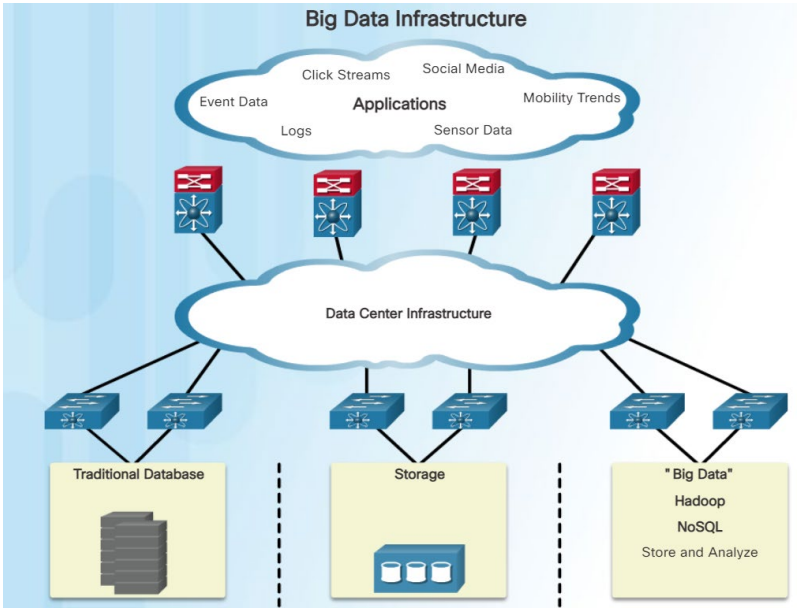
Mnoho společností si uvědomuje smysl investic do některých technologií Big Dat, aby zůstaly konkurenceschopné na svém trhu. Jejich datová infrastruktura může v současnosti vypadat podobně, jako na tomto obrázku. Obvykle je přístup k datům omezen pouze na několik znalých jednotlivců v rámci organizace.



Společnosti rychle směřují k využití technologií Big Data k podpoře business intelligence. Podle NIST (Národní institut standardů a technologií) se **paradigma big dat skládá z distribuce datových systémů napříč horizontálně propojenými nezávislými zdroji**, aby bylo dosaženo **škálovatelnosti** potřebné pro efektivní zpracování rozsáhlých datových sad.

Toto je **horizontální škálovatelnost**. Od vertikální škálovatelnosti se liší v tom, že se nepokouší přidat více výpočetního výkonu, úložiště nebo paměti ke stávajícím strojům. Tyto infrastruktury mohou mnoha uživatelům umožnit bezproblémový a bezpečný přístup k datům současně.

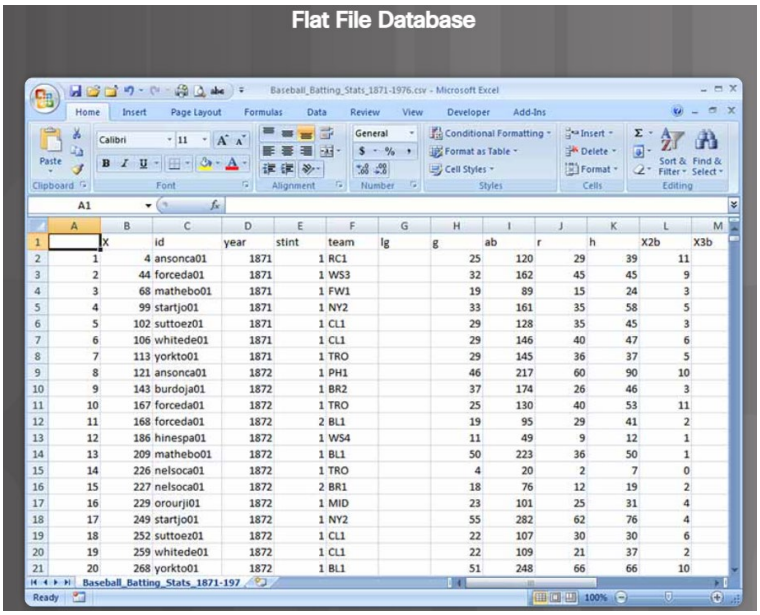
Jedním z takových příkladů mohou být **tisíce online nakupujících nebo mobilních hráčů**. Obrázek představuje infrastrukturu Big dat v organizaci. V tomto příkladu může podnik potřebovat vlastní úložiště, tak Cloud computing.



Základní technologie správy dat

Flat File Databáze

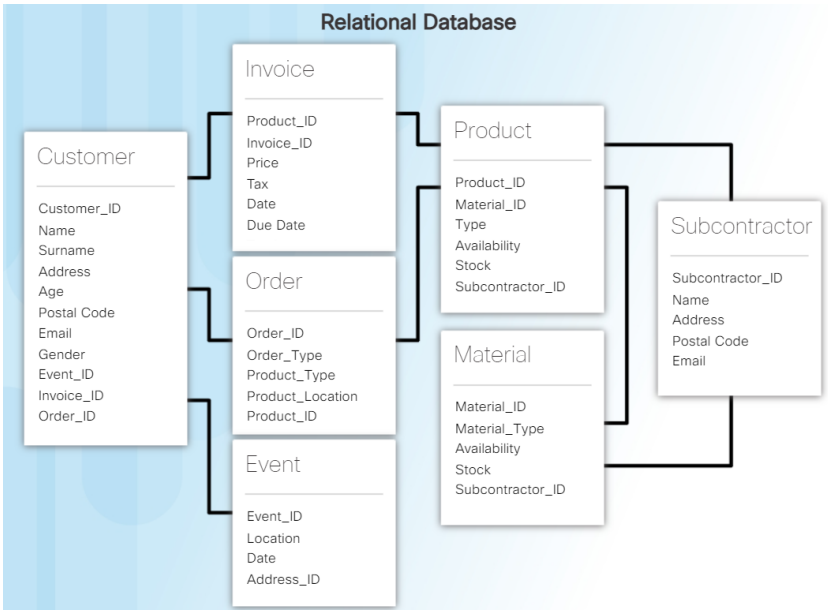
Dříve, než se začalo používat SQL a další databázové programovací jazyky, profesionálové pracovali s tzv. Flat file database (databázemi plochých souborů). Plochá databáze ukládá **záznamy v jediném souboru bez hierarchické struktury**. Jak je znázorněno na obrázku, tyto databáze se skládají ze **sloupců** a **řádků**. Sloupce se také nazývají **atributy** a řádky se také nazývají **záznamy**. Tabulka v Excelu je příkladem ploché databáze.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1		X	id	year	stint	team	lg	g	ab	r	h	X2b	X3b
2	1	4	ansonca01	1871	1	RC1		25	120	29	39	11	
3	2	44	forceda01	1871	1	WS3		32	162	45	45	9	
4	3	68	mathebo01	1871	1	FW1		19	89	15	24	3	
5	4	99	startjo01	1871	1	NY2		33	161	35	58	5	
6	5	102	suttoez01	1871	1	CL1		29	128	35	45	3	
7	6	106	whitede01	1871	1	CL1		29	146	40	47	6	
8	7	113	yorkto01	1871	1	TRO		29	145	36	37	5	
9	8	121	ansonca01	1872	1	PH1		46	217	60	90	10	
10	9	143	burdoja01	1872	1	BR2		37	174	26	46	3	
11	10	167	forceda01	1872	1	TRO		25	130	40	53	11	
12	11	168	forceda01	1872	2	BL1		19	95	29	41	2	
13	12	186	hinespa01	1872	1	WS4		11	49	9	12	1	
14	13	209	mathebo01	1872	1	BL1		50	223	36	50	1	
15	14	226	nelsoca01	1872	1	TRO		4	20	2	7	0	
16	15	227	nelsoca01	1872	2	BR1		18	76	12	19	2	
17	16	229	orourji01	1872	1	MID		23	101	25	31	4	
18	17	249	startjo01	1872	1	NY2		55	282	62	76	4	
19	18	252	suttoez01	1872	1	CL1		22	107	30	30	6	
20	19	259	whitede01	1872	1	CL1		22	109	21	37	2	
21	20	268	yorkto01	1872	1	BL1		51	248	66	66	10	

Relační databázový model, NoSQL

Další generace správy dat se objevila se **systémem správy relačních databází** (RDBMS - Relational Database Management System). Po 30 let to byl standardní přístup ke správě dat. Relační databáze **zachycují vztahy mezi různými soubory dat** a poskytují tak více užitečných informací. Na obrázku jsou tyto vztahy znázorněny čarami. Například podrobnější informace o subdodavatelích, lze získat rozšířením dotazu do databázových tabulek produkty a materiál.



IBM (SQL/DS) a **Oracle** přišli s prvními dvěma komerčními řešeními RDBMS. Většina komerčních řešení RDBMS používá jako svůj dotazovací jazyk **SQL** dodnes. Příklady produktů, které pro přístup k datům používají strukturovaný dotazovací jazyk, zahrnují **MySQL**, **SQLite**, **MS SQL**, **Oracle** a **IBM DB2**.

Další charakteristikou relačních databází je rozdíl mezi databází a systémem používaným pro správu a dotazování do databáze. S RDMS a vlastní databází může obvykle **mnoho uživatelů dotazovat relační databázi současně**. Uživatel normálně nezná všechny vztahy, které existují uvnitř databáze. Uživatel spíše využívá abstraktní pohled na databázi.

Na rozdíl od tradičních systémů pro správu relačních databází SQL, jejichž škálování může být náročné, se nerelační databáze SQL (NoSQL) škálují velmi dobře jako distribuované databáze. Protože NoSQL dokáže zpracovávat velká data a webové aplikace v reálném čase lépe než RDBMS, databázové dotazy NoSQL jsou zaměřeny o shromažďování znalostí, jako jsou informace shromážděné z webových stránek. NoSQL také umožňuje serverovým clusterům zpracovávat data a poskytovat lepší kontrolu nad jejich dostupností. Databáze NoSQL jsou široce přijímány k řešení business procesů.

SQLite

Strukturovaný dotazovací jazyk (SQL) je navržen pro správu, vyhledávání a zpracování dat, včetně velkých dat. SQLite je jednoduchý a snadno použitelný databázový stroj SQL. SQLite je mezi procesová knihovna, která používá samostatný transakční databázový stroj SQL. Kód pro SQLite je ve veřejné doméně, což znamená, že je zdarma k použití pro komerční nebo soukromé účely. SQLite je nejrozšířenější databáze na světě. SQLite používají doslova miliony aplikací s miliardami nasazení. Známý uživatelé SQLite jsou například:

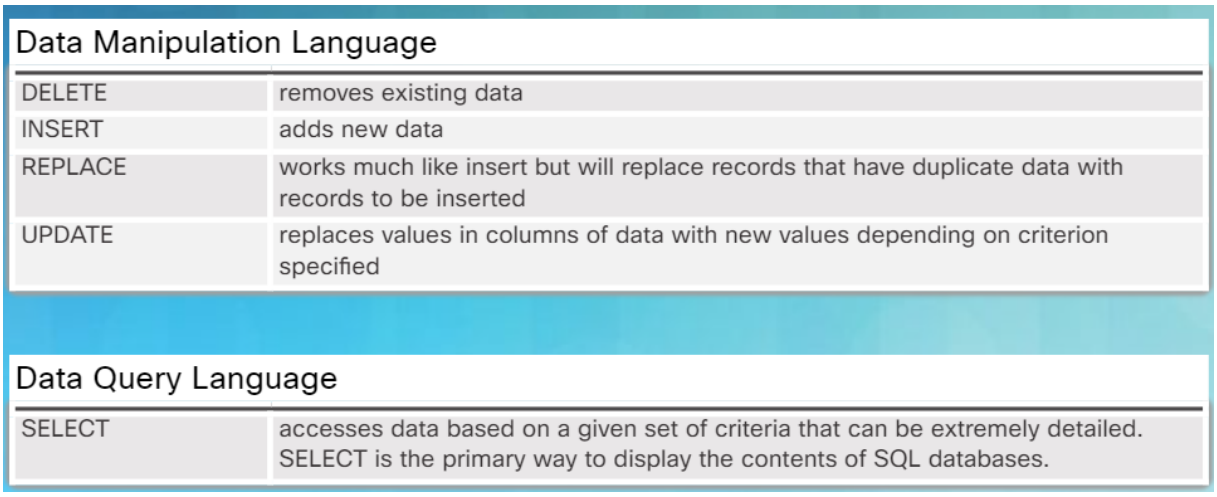
- **Apple** používá SQLite v mnoha nativních aplikací běžících na počítačích a serverech Mac OS-X a na zařízeních iOS, jako jsou iPhony a iPody. SQLite se také používá v iTunes , a to i na hardwaru jiného výrobce než Apple.
- **Adobe** používá SQLite jako formát souboru aplikace pro svůj produkt Photoshop Lightroom . SQLite je také standardní součástí Adobe Integrated Runtime (AIR) . Uvádí se, že Acrobat Reader také používá SQLite.
- Všechny distribuce **Pythonu** od Pythonu 2.5 zahrnují SQLite.
- **Google** používá SQLite ve svém operačním systému pro mobilní telefony Android a ve webovém prohlížeči Chrome .

SQLite má několik užitečných funkcí. Některé z nich jsou uvedeny zde:

- Není vyžadováno žádné nastavení ani správa.
- Má snadno použitelné API.
- Kompletní databáze je uložena v jediném souboru na disku pro různé platformy.
- Lze použít jako formát souboru aplikace.
- Má malou kódovou stopu.
- Jedná se o multiplatformní SQL.
- Podporuje Android, iOS, Linux, Mac, Windows a několik dalších operačních systémů.
- Zdroje pro SQLite jsou ve veřejné doméně.
- Má samostatné rozhraní příkazového řádku (CLI).

SQLite a Python

Data Definition Language	
ALTER TABLE	modifies the structure of an existing table
CREATE DATABASE	creates a new empty database
CREATE TABLE	creates a table within an existing database
DESCRIBE	displays the structure of a table
DROP DATABASE	completely deletes an entire database
DROP TABLE	deletes a table from within a database
USE	opens the database to be worked with



Data Manipulation Language	
DELETE	removes existing data
INSERT	adds new data
REPLACE	works much like insert but will replace records that have duplicate data with records to be inserted
UPDATE	replaces values in columns of data with new values depending on criterion specified

Data Query Language	
SELECT	accesses data based on a given set of criteria that can be extremely detailed. SELECT is the primary way to display the contents of SQL databases.

SQLite je aktuálně ve verzi 3, proto se modul pro práci SQLite jmenuje sqlite3. Jedná se o modul ze standardní knihovny Pythonu, takže by měl být k dispozici vždy.

```
import sqlite3 as sq1

conn = sq1.connect('logins.db')

!csvsql –db sqlite://///logins.db –insert logins.csv

cur = conn.cursor()

query = 'SELECT * FROM logins LIMIT 5'

cur.execute(query)

for row in cur:

        print(row)

cur.close()
conn.close()
```