```
SELECT fid
FROM   friend AS f, update AS u
WHERE  f.uid= uid1 AND f.fid=u.uid AND u.country=UK
```

A Conventional DBMS plan

find all my friends who have online update records from UK

- **friend**(uid, fid)
- **update**(id, uid, country, date, post, ...)

S1: for each **`friend`**, find a set F_1 of all my friends

S2: for each `fid` f_0 in F_1, check whether f_0 has a record in **`update`** from UK

a nested loop

4

A Conventional Query Plan

a linear scan

Facebook:

- billions of **`friend`** tuples
- trillions of **`update`** tuples

Fast! 12GB/s

- S1 takes 20 mins (|F1|=100)
- S2 takes 2000 mins

Polynomial time queries become intractable on big data

A Conventional Query Plan

find all my friends who have online update records from UK

- **friend**(uid, fid)
- **update**(id, uid, country, date, post, ...)

```
SELECT fid
FROM   friend AS f, update AS u
WHERE  f.uid= uid1 AND f.fid=u.uid AND u.country=UK
```

A Conventional DBMS plan

a linear scan

S1: for each **friend**, find a set F_1 of all my friends

a nested loop

S2: for each fid f_0 in F_1, check whether f_0 has a record in **update** from UK

Facebook:

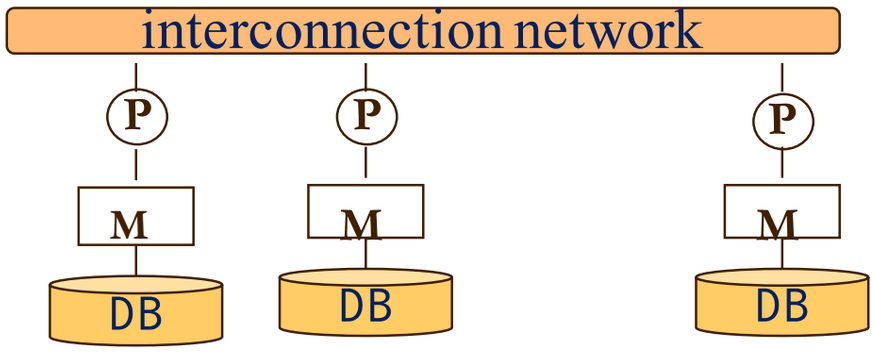
- billions of **friend** tuples

- S1 takes 20 mins (|F1|=100)

Polynomial time queries become intractable on big data

Common Wisdom

- Parallel query processing



Assuming linear scalability, using 50,000 processors

- 2,020 mins is reduced to 2.4s