



# *SQL fyrirspurnarmálið*

framhald af *Gagnasöfn og SQL*

Hjálmtyr Hafsteinsson  
Tölvunarfræði  
Háskóli Íslands

# Efni námskeiðs

- Upprifjun og SQLite
- Tengingar tafla (*join*)
- Hreiðraðar fyrirspurnir (*nested queries*)
- Ytri tengingar (*outer join*)
- Notkun á mengjavirkjum (*set operators*)
- Sýndartöflur (*views*)
- Önnur gagnasafnskerfi (ef tími/áhugi)



# SQLite

- Frítt einfalt venslagagnasafnskerfi
  - Hefur nær allar SQL skipanir
  - Mjög auðvelt í uppsetningu
  - Innan við 1MB að stærð
- Notað mjög víða
  - Innbyggð í Firefox, Chrome, iPhone, Android, Win10, Skype, Photoshop, iTunes, ...

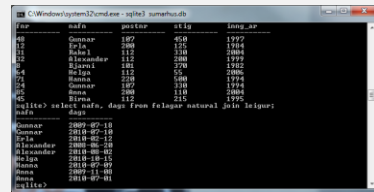
# SQLite

- Náðið í það:  
<http://www.sqlite.org>
- Náðið í sýnisgagnasafn:  
<https://hjalmtyr.github.io/SQL2/>
  - Upplýsingar vegna útleigu á sumarhúsum:

|                 |                           |
|-----------------|---------------------------|
| <b>felagar</b>  | upplýsingar um félagsmenn |
| <b>sumarhus</b> | upplýsingar um sumarhús   |
| <b>leigur</b>   | upplýsingar um leigur     |

# SQLite viðmót

- Getum notað SQLite í gegnum skipanaskel
  - Einfalt í notkun



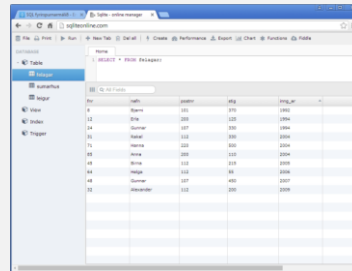
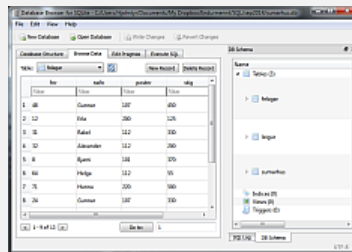
- Til nokkur grafísk viðmót (*GUI*) fyrir SQLite:

- DB Browser

- Bæði Windows og Mac
- Frír og opin hugbúnaður

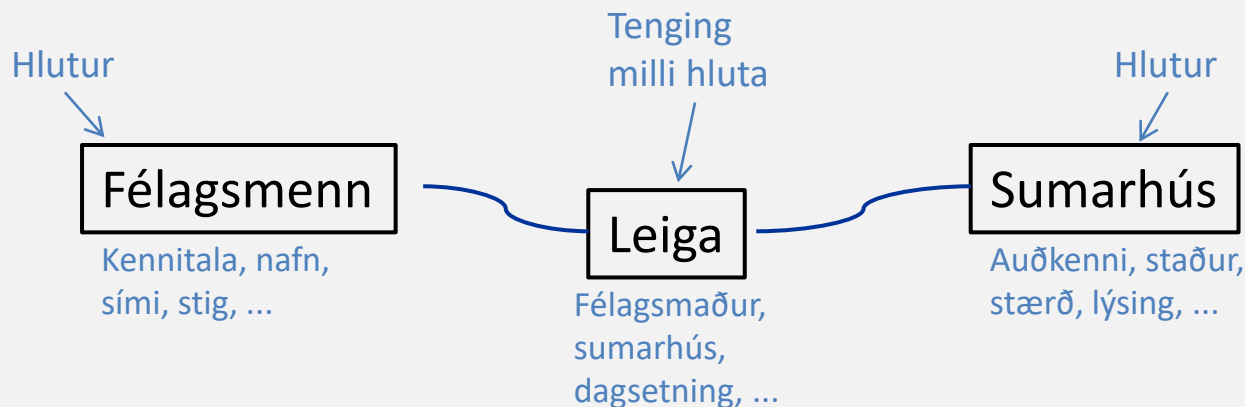
- [sqliteonline.com](http://sqliteonline.com)

- Viðmót sem keyrir í vafra



# Sýnisgagnasafn

- Þrjár töflur með upplýsingum leigu á sumarhúsum



# SQL upprifjun

## Velja línur (**where**)

- Sýnið þá félagsmenn sem hafa minna en 100 stig, í röð eftir stigafjölda?

```
select * from felagar
where stig < 100
order by stig;
```

- Sýnið allar helgarleigur á árinu 2022

```
select * from leigur
where dags like '2022%'
and fj_daga = 3;
```

# SQL upprifjun

## Samsöfnun (**group by**)

- Hver er meðalfjöldi stiga félagsmanna sem búa í Reykjavík?

```
select avg(stig) from felagar  
where postnr < '170';
```

- Sýnið fjölda leiga á hvert sumarhús

```
select husnr, count(*) from leigur  
group by husnr;
```



# Æfingar

- Sýnið félagsmenn í röð eftir póstnúmeri  
(...**order by**...)
- Sýnið þá bústaði sem eru stærri en 60m<sup>2</sup> eða hafa 6 rúm eða fleiri  
(...**where**...)
- Sýnið meðalfjölda stiga eftir inngönguári félaga  
(...**group by**...)

# Tengingar (*join*)

- Hingað til aðeins unnið með eina töflu í einu
- Getum við sett öll gögn í eina töflu?
  - Já, en það hefur galla

*Umfremd (redundancy)*

*Vandræði við breytingar*

*Innsetningar- og eyðingarvandræði*

# Vandamál við eina töflu

- Umfremd (*redundancy*)

| fnr | nafn   | postnr | stig | inng_ar | dags      | fj_daga | husnr |
|-----|--------|--------|------|---------|-----------|---------|-------|
| 48  | Gunnar | 107    | 450  | 2017    | 8.7.2022  | 7       | 1001  |
| 48  | Gunnar | 107    | 450  | 2017    | 16.7.2021 | 7       | 1005  |
| 12  | Erla   | 200    | 125  | 1994    | 13.2.2022 | 3       | 1005  |

- Gildin **postnr**, **stig** og **inng\_ar** eru endurtekin fyrir hverja skráða leigu
- Hvað ef við viljum bæta við mynd af hverjum félagsmanni?

# Vandamál við eina töflu

- Vandræði við breytingar

| fnr | nafn   | postnr | stig | inng_ar | dags      | fj_daga | husnr |
|-----|--------|--------|------|---------|-----------|---------|-------|
| 48  | Gunnar | 107    | 450  | 2017    | 8.7.2022  | 7       | 1001  |
| 48  | Gunnar | 107    | 450  | 2017    | 16.7.2021 | 7       | 1005  |
| 12  | Erla   | 200    | 125  | 1994    | 13.2.2022 | 3       | 1005  |

- Ef stig Gunnar lækka um 50 þá þarf að breyta því á öllum stöðum
- Ef póstnúmerið hjá einum félagsmanni breytist þá þarf að fara í gegnum alla töfluna

# Vandamál við eina töflu

- Innsetningar- og eyðingarvandræði

| fnr | nafn   | postnr | stig | inng_ar | dags      | fj_daga | husnr |
|-----|--------|--------|------|---------|-----------|---------|-------|
| 48  | Gunnar | 107    | 450  | 2017    | 8.7.2022  | 7       | 1001  |
| 48  | Gunnar | 107    | 450  | 2017    | 16.7.2021 | 7       | 1005  |
| 12  | Erla   | 200    | 125  | 1994    | 13.2.2022 | 3       | 1005  |

- Getum ekki bætt inn félagsmanni ef hann hefur ekki leigt sumarhús ennþá
- Getum ekki eytt út síðustu leigu félagsmanns því þá tapast allar aðrar upplýsingar um hann

# Lausn á vandræðum

- Bjótum töfluna upp í tvær töflur
  - Önnur aðeins með upplýsingar um félagsmenn
  - Hin aðeins með upplýsingar um leigur

| fnr | nafn   | postnr | stig | inng_ar |
|-----|--------|--------|------|---------|
| 48  | Gunnar | 107    | 450  | 2017    |
| 12  | Erla   | 200    | 125  | 1994    |
| 31  | Rakel  | 112    | 330  | 2014    |

| fnr | dags      | fj_daga | husnr |
|-----|-----------|---------|-------|
| 48  | 8.7.2022  | 7       | 1001  |
| 48  | 16.7.2021 | 7       | 1005  |
| 12  | 13.2.2022 | 3       | 1005  |

Að vísu **fnr** í báðum töflum, en  
þurfum það til að tengja þær saman

# Tengingar

- Þurfum nú að tengja töflurnar saman
  - Ein leið:

```
select nafn, dags
  from felagar, leigur
 where felagar.fnr = leigur.fnr;
```

- Jafngild leið (nýrri útgáfa):

```
select nafn, dags
  from felagar join leigur
    on felagar.fnr = leigur.fnr;
```

# Merking tengingar

```
SELECT dálkar  
      FROM tafla1 JOIN tafla2  
      ON skilyrði;
```

Fyrir allar mögulegar samsetningar á línunum úr **tafla1** og **tafla2**

Ef skilyrði er **satt**, þá sýna dálka úr þeirri samsetningu



# Nokkur dæmi

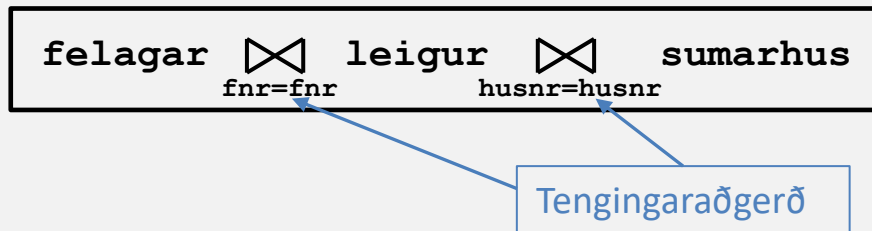
```
select *  
  from felagar join leigur  
    on felagar.fnr = leigur.fnr;
```

```
select nafn, dags, husnr  
  from felagar join leigur  
    on felagar.fnr = leigur.fnr  
 where dags >= '2022-01-01';
```

# Tenging fleiri tafla

- Getum tengt saman margar töflur

```
select nafn, stadur, dags
  from felagar join leigur
        on felagar.fnr = leigur.fnr
      join sumarhus
        on leigur.husnr = sumarhus.husnr
 where fj_daga < 7;
```



# Nokkur dæmi

- Finna nöfn sumarhúsa sem Gunnar hefur leigt

```
select stadur
  from felagar join leigur
    on felagar.fnr = leigur.fnr
  join sumarhus
    on leigur.husnr = sumarhus.husnr
 where nafn = 'Gunnar';
```

# Einfölduð tenging

- Ef samanburður er = og dálkar heita sömu nöfnum þá hægt að nota náttúrulega tengingu

```
select stadur
      from felagar natural join leigur
      natural join sumarhus
      where nafn = 'Gunnar' ;
```

# Tenging með samsöfnun

- Finna heildarfjölda leigudaga hjá Önnu

```
select sum(fj_daga)
  from felagar natural join leigur
 where nafn = 'Anna';
```

- Finna meðalfermetrafjölda eftir félagsmönnum

```
select nafn, avg(fermetrar)
  from felagar natural join leigur
      natural join sumarhus
 group by nafn;
```

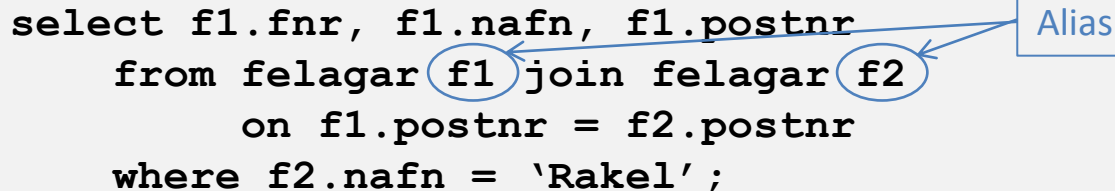
# Æfingar

- Sýnið alla þá sem hafa fengið leigðan bústað 1001
- Sýnið alla þá sem hafa fengið leigðan bústaðinn “Laugarvatn 2”
- Sýnið nafn og fjölda útleiga fyrir hvern bústað
  - Viðbót: ... á árinu 2022
- Fyrir hvern bústað sýnið nöfn þeirra félagsmanna sem hafa fengið hann leigðan í vikuleigu

# Sjálf tenging

- Getum tengt töflu við sjálfa sig
  - Notum þá tvö eintök af töflunni með sitthvoru nafninu (*alias*)
- Finna alla sem búa í sama póstnúmeri og Rakel

```
select f1.fnr, f1.nafn, f1.postnr
  from felagar f1 join felagar f2
    on f1.postnr = f2.postnr
 where f2.nafn = 'Rakel' ;
```



En ef við viljum ekki fá Rakel með í úttakið?

# Sjálf tenging

- Algengustu not á sjálf tengingu er þegar það er innbyrðistenging á milli dálka

| stmnr | nafn  | yfirmadur |
|-------|-------|-----------|
| 2     | Palli | 5         |
| 4     | Gunna | 8         |
| 5     | Ari   | 8         |
| 7     | Anna  | 5         |
| 8     | Sigga | 8         |

Palli hefur Ara sem yfirmann

Sigga hefur engan yfirmann



# Sjálf tenging

- Finna nöfn allra ásamt nöfnum yfirmanna þeirra

```
select s.nafn as stm, y.nafn as yfirm
      from starfsm s join starfsm y
      on y.stmnr = s.yfirmadur;
```

Gefum dálkunum heiti

# Hreiðraðar fyrirspurnir

*(nested queries)*

- Getum notað fyrirspurn inni í annari fyrirspurn
  - Undirfyrirspurnin getur komið á nokkrum stöðum
  - Algengast er að hún komi í **where**-hluta

```
select count(*) from leigur
  where fnr in
    (select fnr from felagar
     where nafn='Gunnar' );
```

Finna fjölda leiga sem Gunnar hefur fengið

# Undirfyrirspurnir *(subqueries)*

- Fyrirspurnir skila annaðhvort einu gildi:

```
select avg(stig) from felagar;
```

eða mengi staka:

```
select nafn from felagar  
where inng_ar < 2000;
```

Getum nota útkomuna í öðrum fyrirspurnum

# Samsöfnun skilar einu gildi

- Finna félagsmann með mesta fjölda stiga

```
select nafn, stig from felagar
where stig = (select max(stig)
              from felagar);
```

Af hverju ekki bara:

```
select nafn, max(stig) from felagar;
```

Flest gagnasafnskerfi gefa villu  
hér, en SQLite skilar "réttu" svari!

# Útkoman er eitt gildi

- Finna þá sem búa í sama póstnúmeri og Helga
  - Hugmynd: Finna fyrst póstnúmer Helgu og finna síðan þá sem hafa það póstnúmer

```
select * from felagar
  where postnr = (select postnr from felagar
                  where nafn = 'Helga');
```

Kemur Helga líka með?

Hvað ef margar niðurstöður?

# Útkoman er eitt gildi

- Finna alla aðra sem búa í sama póstnúmeri og Helga
  - Eins og áður, en viljum ekki fá Helgu

```
select * from felagar
      where nafn <> 'Helga' and
             postnr = (select postnr from felagar
                       where nafn = 'Helga');
```

# Fleiri dæmi með einni útkomu

- Finna þá sem hafa fleiri stig en félagsmaður númer 85

```
select nafn, stig from felagar
      where stig > (select stig from felagar
                    where fnr = 85);
```

- Ef fleiri en eitt gildi þá samanburður við fyrsta

```
select nafn, stig from felagar
      where stig > (select stig from felagar
                    where nafn = 'Gunnar');
```

Sum gagnasafnskerfi gefa villu hér

# Æfingar

- Sýnið nafnið á sumarhúsinu með mestan fjölda rúma
- Sýnið öll sumarhús sem eru minni en Reykir
- Sýnið alla félagsmenn sem búa í öðru póstnúmeri en Bjarni
- Sýnið þá félagsmenn sem hafa leigt oftast en Erla



# Útkoman er mengi

- Getum þá athugað hvort tiltekið gildi sé í menginu
  - Finna alla félagsmenn sem hafa leigt sumarhús:

```
select * from felagar
      where fnr in (select fnr from leigur);
```

eða alla sem hafa leigt á ákveðnu tímabili:

```
select * from felagar
      where fnr in (select fnr from leigur
                    where dags between '2022-06-01'
                                     and '2022-08-31');
```

# Fleiri dæmi

- Finna nöfn þeirra sem hafa leigt bústaðinn Reykir

```
select nafn from felagar
where fnr in
    (select fnr from leigur
     where husnr in
        (select husnr from sumarhus
         where stadur = 'Reykir'));
```

Einnig hægt að gera þessa  
fyrirspurn með tengingu (*join*)

# Útkoman er mengi

- Getum líka athugað hvort gildi sé ekki í menginu
  - Finna þau sumarhús sem aldrei hafa verið leigð

```
select * from sumarhus
      where husnr not in (select husnr
                          from leigur);
```

# Gagnvísandi fyrirspurn

(*correlated query*)

- Oft er undirfyrirspurn háð skipuninni sem inniheldur hana
  - Finna félagsmenn sem hafa 2 leigur

```
select * from felagar f
  where 2 = (select count(*) from leigur l
             where f.fnr = l.fnr);
```

Nafnið **f** í undirfyrirspurn á  
við **f**-ið í aðalfyrirspurninni

# Fleiri dæmi

- Getum athugað hvort undirfyrirspurn skili einhverju með **exists**

Sýna þá sem hafa fengið bústaði á leigu

```
select fnr, nafn from felagar f
      where exists (select * from leigur l
                    where f.fnr = l.fnr);
```

# Flóknari dæmi

- Finna hvort einhverjir tveir félagsmenn hafi sama nafn
  - **Hugmynd:** Fyrir alla félagsmenn: Er til félagsmaður sem hefur sama nafn, en annað númer? Ef svo er þá skrifum við hann út

```
select nafn from felagar f1
  where exists (select fnr from felagar f2
                where f1.nafn = f2.nafn
                and f1.fnr <> f2.fnr);
```

# Fyrirspurnir í **from**-hluta

- Undirfyrirspurn í **from**-hluta er eins og tafla
  - Oftast til betri leiðir til að gera þessar fyrirspurnir

```
select fnr, nafn, fj.fjoldi
      from felagar natural join
      (select fnr, count(*) fjoldi
       from leigur
       group by fnr) fj
where fj.fjoldi > 1;
```

Skilgreinir töflu með **fnr** og fjölda leiga fyrir hvern félagsmann

# Fyrirspurnir í `select`-hluta

- Finna þá sem eru með fleiri stig en meðaltalið

```
select nafn, stig from felagar
where stig > (select avg(stig)
              from felagar);
```

og ef við viljum fá meðalstigin með:

```
select nafn, stig, (select avg(stig)
                    from felagar) medal
from felagar
where stig > medal;
```

Má aðeins skila  
einu gildi



# Fyrirspurnir í `select`-hluta

- Sýna nafn og prósentu stiga miðað við mesta fjölda stiga

```
select nafn,  
       stig,  
       stig*100.0/(select max(stig)  
                   from felagar) as prosent  
from felagar;
```



Finum mesta fjölda stiga  
til að reikna prósentu

# Aðrar SQL skipanir

- Getum notað undirfyrirspurnir í öðrum SQL skipunum

Hækka stigin hjá öllum þeim sem leigðu tvisvar eða oftar á árinu 2022

```
update felagar set stig = stig + 10
  where 2 <= (select count(*) from leigur l
              where days like '2022%' and
              l.fnr = felagar.fnr);
```

# Undirfyrirspurnir, samantekt

- Geta verið á ýmsum stöðum:
  - Í **where**-hluta
    - Ef útkoma mengi: notum **in** eða **exists**
    - Ef útkoma eitt gildi: notum samanburði (=, <, >, <=, ...)
  - Í **from**-hluta
    - Þurfum þá að gefa aukanafn (*alias*)
    - Hegða sér annars eins og töflur
  - Í **select**-hluta
    - Mega þá aðeins skila einu gildi

# Æfingar

- Sýnið þá félagsmenn sem aldrei hafa fengið sumarhús á leigu
- Sýnið þau sumarhús sem hafa verið leigð sjaldnar en tvisvar
- Sýnið þá félagsmenn sem aldrei hafa leigt sér sumarhús með 10 rúmum

# Ytri tengingar

- Í venjulegri tengingu þurfa línur í töflunum að passa saman
- Viljum stundum líka fá línur sem ekki passa við neina línu í hinni töflunni
  - Fá lista yfir alla félagsmenn og bústaði sem þeir hafa leigt
- Notum þá ytri tengingu (*outer join*)

# Vinstri ytri tenging

- Tengjum saman **felagar** og **leigur**

```
select nafn, husnr, dags
  from felagar f left join leigur l
    on f.fnr = l.fnr;
```

Þeir félagsmenn sem ekki hafa neinar leigur hafa **NULL** í dálkum úr **leigur**-töflunni

| nafn      | husnr | dags       |
|-----------|-------|------------|
| -----     | ----- | -----      |
| Gunnar    | 1001  | 2022-07-08 |
| Gunnar    | 1005  | 2021-07-16 |
| Erla      | 1005  | 2022-02-13 |
| Rakel     | NULL  | NULL       |
| Alexander | 1004  | 2020-06-20 |
| Alexander | 1004  | 2022-08-05 |
| Bjarni    | NULL  | NULL       |
| ...       |       |            |

# Hægri ytri tenging

- Þá eru notuð öll stökin úr hægri töflunni
  - Sýna öll sumarhús og leigur þeirra

```
select stadur, fnr, dags  
  from leigur l right join sumarhus s  
    on l.husnr = s.husnr;
```

Vandamál! SQLite styður ekki hægri tengingu  
Lausn: Snúum þá röð taflanna við!

```
select stadur, fnr, dags  
  from sumarhus s left join leigur l  
    on l.husnr = s.husnr;
```

# Full ytri tenging

- Notar öll stök úr báðum töflum, fyllir upp í svið með **NULL** ef samsvarandi gildi vantar
- Ekki eins nytsöm aðgerð og vinstri (eða hægri) tenging
- Mörg gagnasafnskerfi styðja ekki þessa aðgerð beint, t.d. SQLite
  - Hægt að útfæra hana með öðrum aðgerðum (sjá síðar)



# Náttúruleg ytri tenging

- Getum líka sleppt **on**-hlutanum ef dálkarnir heita sama nafni og samanburður er =

```
select nafn, husnr, dags  
      from felagar natural left join leigur;
```

# Finna það sem vantar

- Hægt að nota ytri tengingu til að finna gildi sem vantar
  - Hvaða félagsmenn hafa aldrei pantað sumarhús?

```
select nafn
  from felagar natural left join leigur
 where husnr is NULL;
```

# Margar tengingar

- Röð tenginga getur skipt máli
- Berið saman niðurstöður þessara skipana:

```
select nafn, stadur, fj_ruma  
  from felagar natural left outer join leigur  
                                natural join sumarhus;
```

```
select nafn, stadur, fj_ruma  
  from felagar natural left outer join (leigur  
                                natural join sumarhus);
```

# Æfingar

- Sýnið alla félagsmenn sem búa í Reykjavík og þær leigur sem þeir eiga
- Sýnið alla félagsmenn og fjölda daga sem þeir hafa leigt sumarhús
- Sýnið þau sumarhús sem aldrei hafa verið leigð (notið ytri tengingu)

# Mengjaaðgerðir

- Töflur eru mengi af línum
  - Getum því notað mengjaaðgerðir á þær
- SQL hefur þrjár mengjaaðgerðir:
  - Sammengi (*union*)
  - Sniðmengi (*intersection*)
  - Mengjamunur (*except*)
- Í mengjaaðgerðunum er tvítekningum sjálfkrafa eytt
  - Ekki gert í öðrum SQL skipunum

# Sammengi

- Sameinum útkomur tveggja fyrirspurna

```
select fnr, nafn from felagar
      where stig > 400
union
select fnr, nafn from felagar
      where postnr < '112';
```

Útkomur fyrirspurnanna þurfa að hafa sama fjölda dálka og þeir þurfa að vera af sömu gerð

# Sniðmengi

- Sýna þær línur sem eru í báðum útkomum

```
select fnr, nafn from felagar
      where stig > 400
intersect
select fnr, nafn from felagar
      where postnr < '112';
```

# Aðrar útfærslur

- Oftast hægt að nota **or** í stað sammengis og **and** í stað sniðmengis

```
select fnr, nafn from felagar  
      where stig > 400 or  
            postnr < '112';
```

- Þetta gengur þó ekki alltaf



# Dæmi um sammengi

```
select fnr from leigur  
  where dags like '2021%' or  
         dags like '2022%';
```

## Jafngilt

```
select fnr from leigur  
  where dags like '2021%'  
union  
select fnr from leigur  
  where dags like '2022%';
```

# Dæmi um sniðmengi

- Hvað ef við viljum fá þá sem hafa leigt bæði á árinu 2021 og 2022?
  - Setjum **intersect** í stað **union** eða **and** í stað **or**:

```
select fnr from leigur
  where dags like '2021%' and
         dags like '2022%';
```

and hér í stað or

... en, hvaða dagsetningar eru  
bæði á árinu 2021 og 2022?!

# Dæmi um sniðmengi

- Getum leyst þetta með því að tvítaka töfluna

```
select l1.fnr from leigur l1, leigur l2
where l1.fnr = l2.fnr and
      l1.dags like '2021%' and
      l2.dags like '2022%';
```

Skýrara að nota  
**intersect**




```
select fnr from leigur
where dags like '2021%'
intersect
select fnr from leigur
where dags like '2022%';
```

# Útfærsla á fullri ytri tengingu

- Sammengi vinstri og hægri tengingar er full tenging

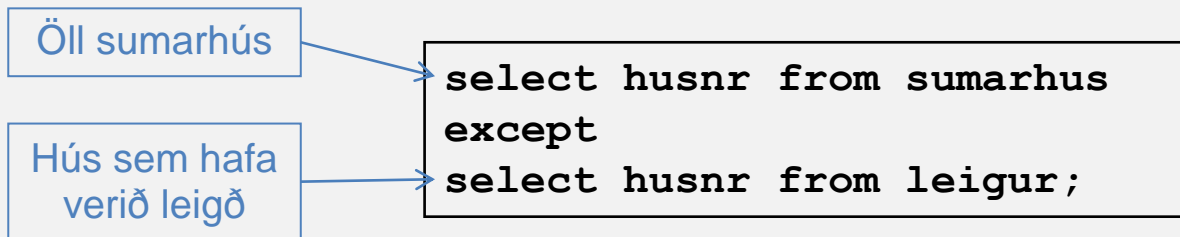
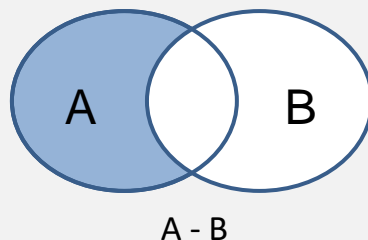
```
select nafn, husnr, dags
  from felagar natural left join leigur
union
select nafn, husnr, dags
  from leigur natural left join felagar;
```



SQLite styður ekki hægri tengingu, svo við útfærum hana með vinstri tengingu og viðsnúning á töflum

# Mengjamunur

- Þær línur sem eru í fyrri útkomu en ekki í þeirri seinni



# not in í stað mengjamunar

- Finna alla sem ekki hafa leigt í júlí

```
select fnr from felagar
except
select fnr from leigur
      where dags like '%-07-%';
```

Jafngilt

```
select fnr from felagar
      where fnr not in
            (select fnr from leigur
              where dags like '%-07-%');
```

# Æfingar

- Sýnið félagsnúmer þeirra sem búa í Grafarvogi eða hafa leigt oftari en einu sinni
- Sýnið þá félagsmenn sem hafa leigt bæði hús númer 1001 og 1002
- Sýnið öll sumarhús sem ekkert voru leigð árið 2022

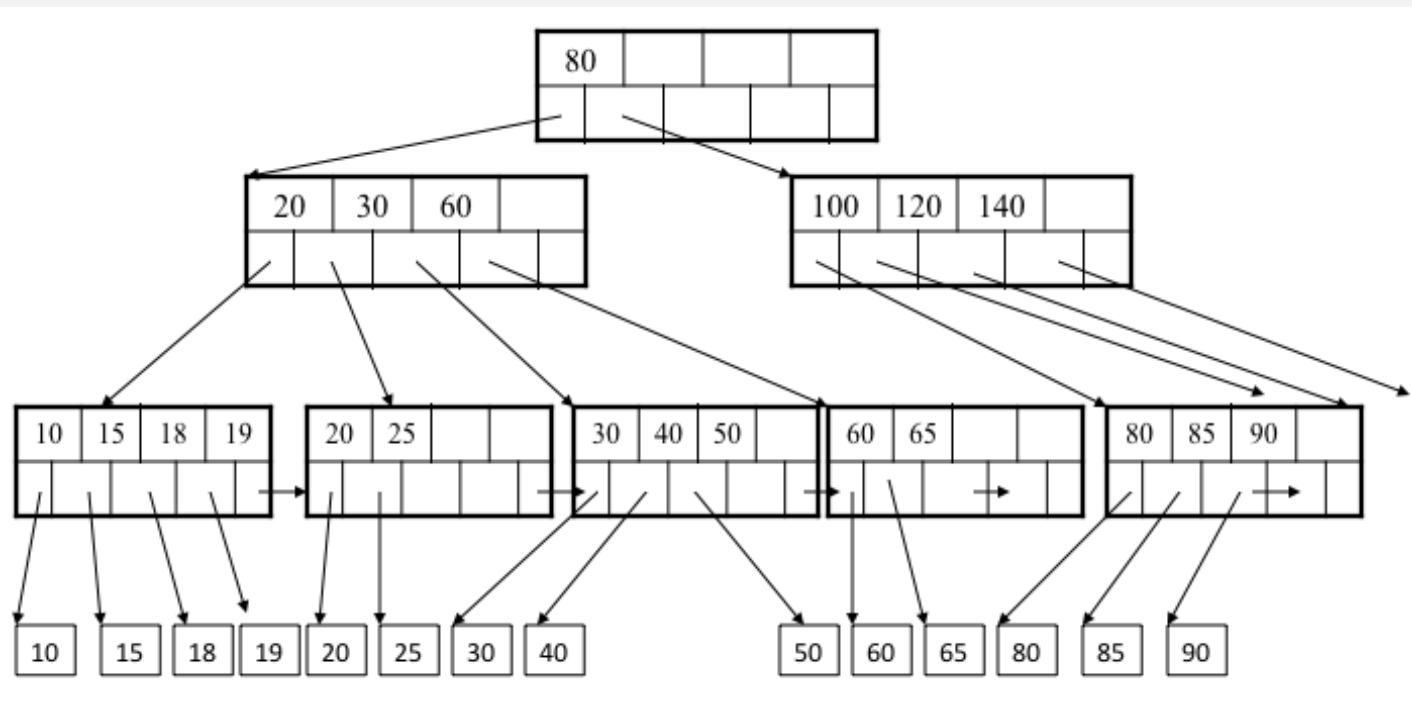
# Vísar

- Getum skilgreint vísa (*indices*) á dálka til að auka leitarhraða
  - Segjum að eftirfarandi fyrirspurn sé algeng:

```
select * from felagar
where nafn = 'Helga';
```
  - Kerfið leitar að þessu með því að fara línu fyrir línu í gegnum alla töfluna
  - Getum búið til leitartré, **B-tré**, sem leyfir okkur mun hraðari leit að nafninu




# B-tré




# Dæmi um vísi

- Búum til vísi á dálkinn **nafn** í **felagar**

```
create index nafn_idx on felagar(nafn);
```



Nafnið á vísinum



Mega vera margir dálkar hér

- Líka til einkvæmir vísar:

```
create unique index fnr_idx on felagar(fnr);
```



Þá má ekki setja tvítekin gildi í þennan dálk

# Notkun á vísum

- Vísar eru ekki ókeypis
  - Við hverja innsetningu, eyðingu eða breytingu á vísagildi þarf að uppfæra vísinn
- Vísar borga sig aðeins ef
  - mikið magn gagna
  - oft leitað að dálknum sem vísirinn er á
- Ekki alltaf ljóst hvaða vísa ætti að búa til
  - Prófa sig áfram og tímamæla

# Sýndartöflur (*views*)

- Sýndartöflur eru niðurstöður úr fyrirspurn
  - Innihalda engin gögn
  - Eru búnar til þegar þörf er á þeim
- Oft notaðar til að gefa aðra sýn á gagnasafnið
  - Fela tilteknar línur
    - Til dæmis starfsmannatafla án yfirmanna
    - Einkunnatafla með eingöngu þínum einkunnum
  - Einfalda fyrirspurnir
    - Búa til sýndartöflu sem tengir allar töflur saman

# Dæmi um sýndartöflur

- Búa til sýndartöflu með úrvalsfélögum:

```
create view vip as
  select * from felagar
  where stig > 400 and
         inng_ar <= 2000;
```

- Getum svo unnið með þessa töflu eins og hverja aðra töflu
  - Inniheldur sömu dálka og **felagar**, en færri línur

# Dæmi um sýndartöflur

- Fela ýmsar upplýsingar um félaga

```
create view fel_post as  
  select fnr, nafn, postnr  
  from felagar;
```

Leyfum sumum notendum aðeins  
að sjá póstnúmer félagsmanna, en  
ekki aðrar upplýsingar

# Dæmi um sýndartöflur

- Getum notað til að einfalda fyrirspurnir:

```
create view allt as
  select * from felagar natural join
         leigur natural join
         sumarhus;
```

Getum nú gert:

```
select nafn, stadur from allt
  where dags >= '2022-01-01';
```

# Dæmi um sýndartöflur

- Getum notað til að setja gögn betur fram

```
create view movieform as
  select m.id,
         title || ' (' || year || ')' as title,
         score,
         name as director
  from movie m join actor on director = actor.id;
```

Búum til nýjan titill  
með ártalinu aftast

Fáum nú nafn  
leikstjórans með



# Kostir sýndartafla

- Það verður engin umfremd þó við búum til sýndartöfluna **allt**
  - Það eru engin gögn í henni!
- Hægt að gefa mismunandi notendum aðgang að mismunandi sýndartöflum
  - Sjá bara það sem þeir mega sjá
  - SQLite hefur ekki skilgreinda notendur

# Breytingar á sýndartöflum

- SQL staðallinn segir að það eigi að vera hægt að breyta gildum í sýndartöflum
  - Hægt að nota **insert**, **update** og **delete**
- Alls ekki auðvelt, þarf að snúa við fyrirspurninni sem skilgreinir sýndartöfluna
- Flest "stóru" gagnasafnskerfið ráða við að breyta sýndartöflum
  - SQLite leyfir það ekki!

# Æfingar

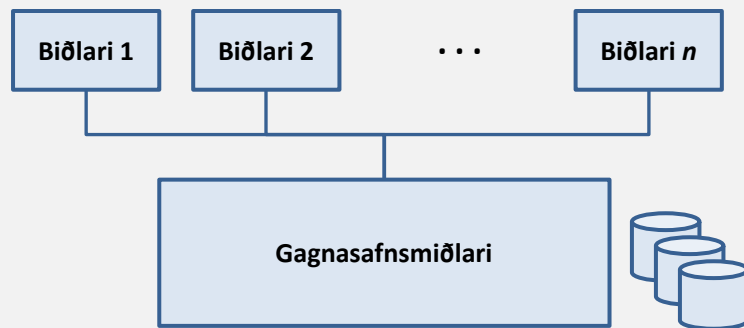
- Búið til sýndartöfluna **allt** og sýnið nöfn þeirra félagsmanna sem hafa leigt bústaðinn "Laugarvatn 1"
- Búið til sýndartöflu sem hefur aðeins dálkana **fnr**, **nafn** og **inng\_ar** og inniheldur aðeins þá sem búa utan Reykjavíkur

# Önnur gagnasafnskerfi

- Margar gerðir gagnasafnskerfa:
  - Stór biðlara-miðlara kerfi (*client-server*)
    - Oracle, DB2, SQL Server, Informix, ...
  - Frí millistór biðlara-miðlara kerfi
    - MySQL, PostgreSQL, Firebird ...
  - Lítil einstaklingskerfi
    - SQLite, MS Access, InterBase, Apache Derby ...
  - Stór, sérhæfð gagnasafnskerfi
    - MongoDB, Cosmos DB, ...

# Miðlara-biðlara kerfi

- Gagnasafnsmiðlari keyrir á sérstakri tölvu
  - Margir biðlarar með aðgang á sama tíma
  - Sumir biðlarar á nærneti, aðrir yfir Internet



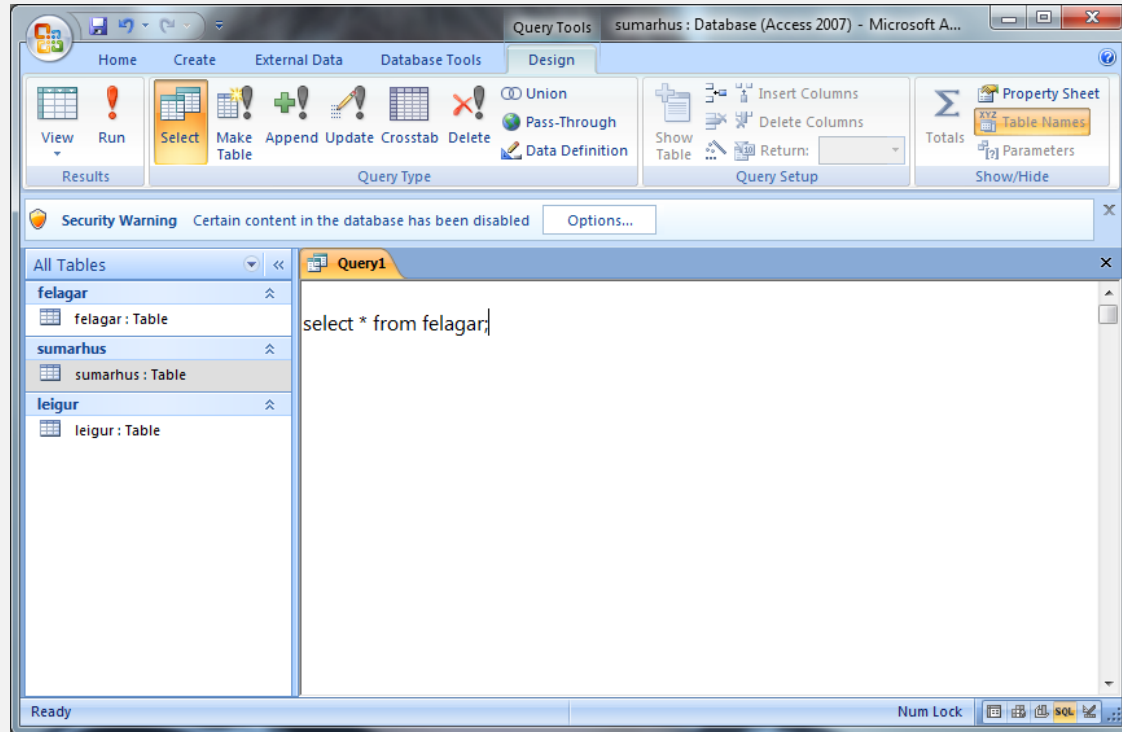
# Aðrir eiginleikar

- Stór gagnasafnskerfi hafa ýmsa aðra eiginleika
  - Innbyggð vefþjónusta (*web services*)
  - Vöruhús gagna (*data warehousing*)
  - Dreifð gagnasöfn (*distributed databases*)
  - Öflug bestun fyrirspurna (*query optimization*)
  - Gagnaöryggi (*data security*)
  - og margt fleira

# MS Access

- Getum notað SQL í MS Access
  - Fara í **Create** valmynd
  - Smella á **Query Design** hnapp
  - Loka strax glugganum sem kemur upp
  - Smella á niðurör á **View** hnappi og velja **SQL View**
  - Slá inn SQL skipun og smella á **Run** hnapp

# SQL í MS Access





# SQL í MS Access

- Útfærir ekki allan SQL staðal
  - Engin náttúruleg tenging (*natural join*)
  - Verður að skrifa **inner join**, ekki nóg að skrifa **join**
  - Útfærir **union**, en ekki **intersect** eða **except**
- Hefur ýmsar viðbætur
  - Öflugri reglulegar segðir (**like**-samanburður)
- Ætlast til að fyrirspurnir séu búnar til sjónrænt (í *Query Design*)

# Hvað næst?

- Meira um SQL:
  - Ytri lyklar (*foreign keys*)
  - Innri föll (*stored procedures*), kveikjur (*triggers*)
- Meira um gagnasöfn:
  - Hönnun gagnasafna (*database design*)
  - Notendur og réttindi (*privileges*)
  - Hreyfingar (*transactions*) og samskeiða vinnsla (*concurrency control*)
  - Aðrar gerðir gagnasafnskerfa (*NoSQL*)