

Unison Tutorial

Reece Hart

2006-05-12

Tutorial Outline

- Introduction
 - data sources, algorithms, update scheme
- Schema
 - overview, design themes, critical tables
- Access
 - web pages, command line tools, perl API, psql
- Example Queries
 - Finding sequences
 - Finding parameters
 - Getting predictions for a sequence
 - Mining for sequence based on predictions
 - Tips
- Future Plans

What Can I Do With Unison?

- Retrieve sequence analysis for a single sequence.
- Mine for sequences based on predicted features, sequence origins, taxonomy, patents, orthology, and structure.
- Find all sources of a single sequence.
- Find patents for a sequence.
- Locate sequence variations relative to domains and in structure.
- Build new tools.

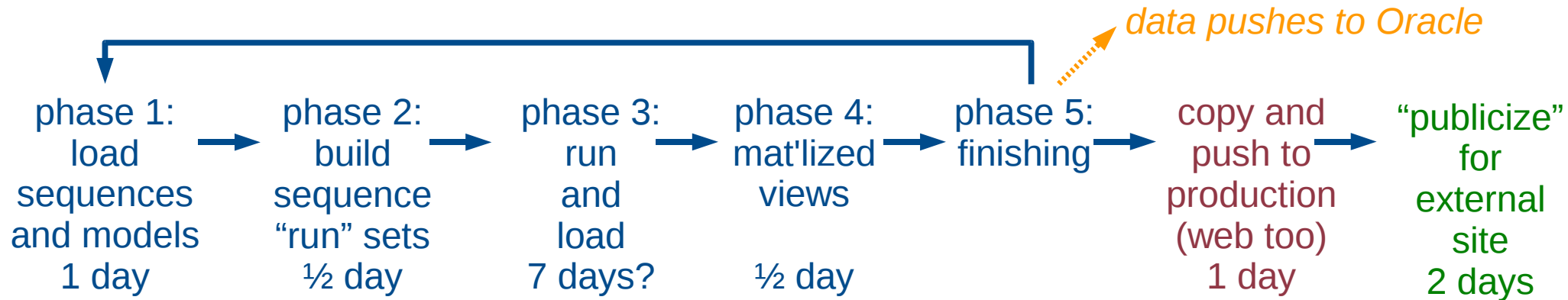
Design Goals

- Sequences are stored non-redundantly.
 - eliminates redundant computation and analysis
 - Results are keyed to sequences, parameters, and optionally a model.
 - Sequences are immutable and therefore results are never stale.
 - Sequences are linked to their origins and aliases.
- Fast, reliable, differential updates.
- Multiple result sets for different invocations
- Make no assumptions and provide no interpretations.
- Synopses of prediction results only, but and enable regeneration of results.

Unison Contents

- Non-redundant Sequences
 - UniProtKB/Swiss-Prot, IPI, Genengenes, Genehub representative sequences, RefSeq, Curagen, Incyte, ..., Ensembl *ab initio*, miscellaneous fragments
- Non-redundant Results
 - Pfam, TMHMM, SignalP, protcomp
 - BIG-PI, PSI-PRED, RegExp motifs
 - disprot, dispro, pmap
- Lots of other Data
 - patents, PDB, SCOP, GO, GOng, NCBI tax, HomoloGene, MINT, ...
- Statistics
 - 75 tables, 108 views, 120 functions
 - ~6 CPU-years' worth of data, >440M protein features
 - 14GB of compressed data, 130GB on disk w/indexes

Update Procedure and Run Sets



Set	Criteria	Algorithms
runA	human 100-1000 AA reliable origins	prospect antigenic BLAST†
runB	human, mouse, rat 100-1500 AA reliable origins	Pfam (fs & ls) BIG-PI RegExp PSIPRED
runC	human, mouse, rat, cow, zebrafish 100-3000 AA all sequence sources	pepcoil SignalP TMHMM antigenic pmap protcomp
	<i>ad hoc</i>	disprot† dispro†

† these methods are currently unsupported

Implementation

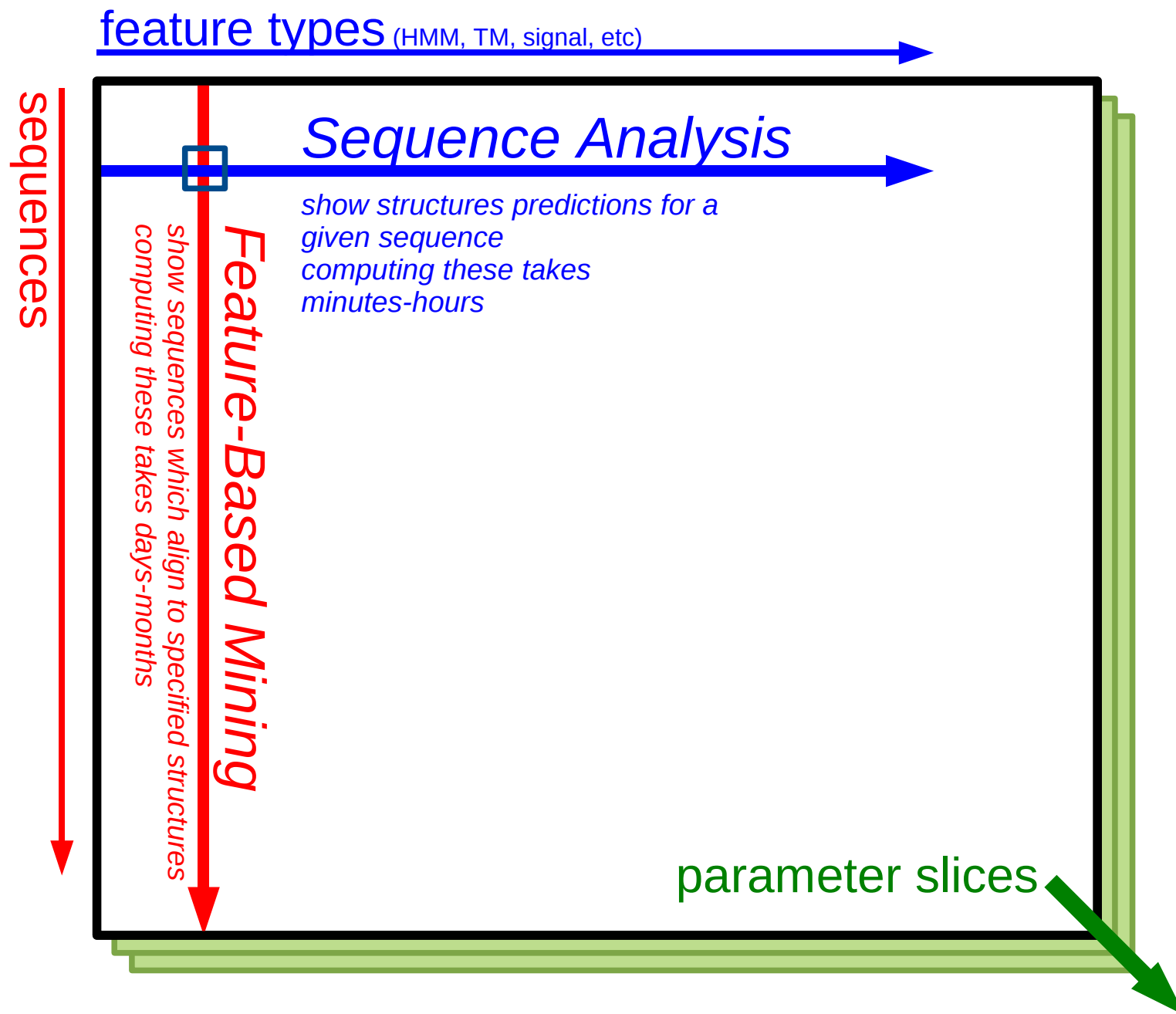
- Hardware
 - hostname csb
 - 4 dual-core Opterons, 2.4 GHz
 - 32GB RAM
 - 500GB FC-RAID
- Linux
 - SuSE 10.0, kernel 2.6
- PostgreSQL 8.1.3
 - 3 databases: csb, csb-stage, csb-dev
 - unison is a schema within each
- Perl 5.8
- Apache 2.0 web pages

Unison Schema

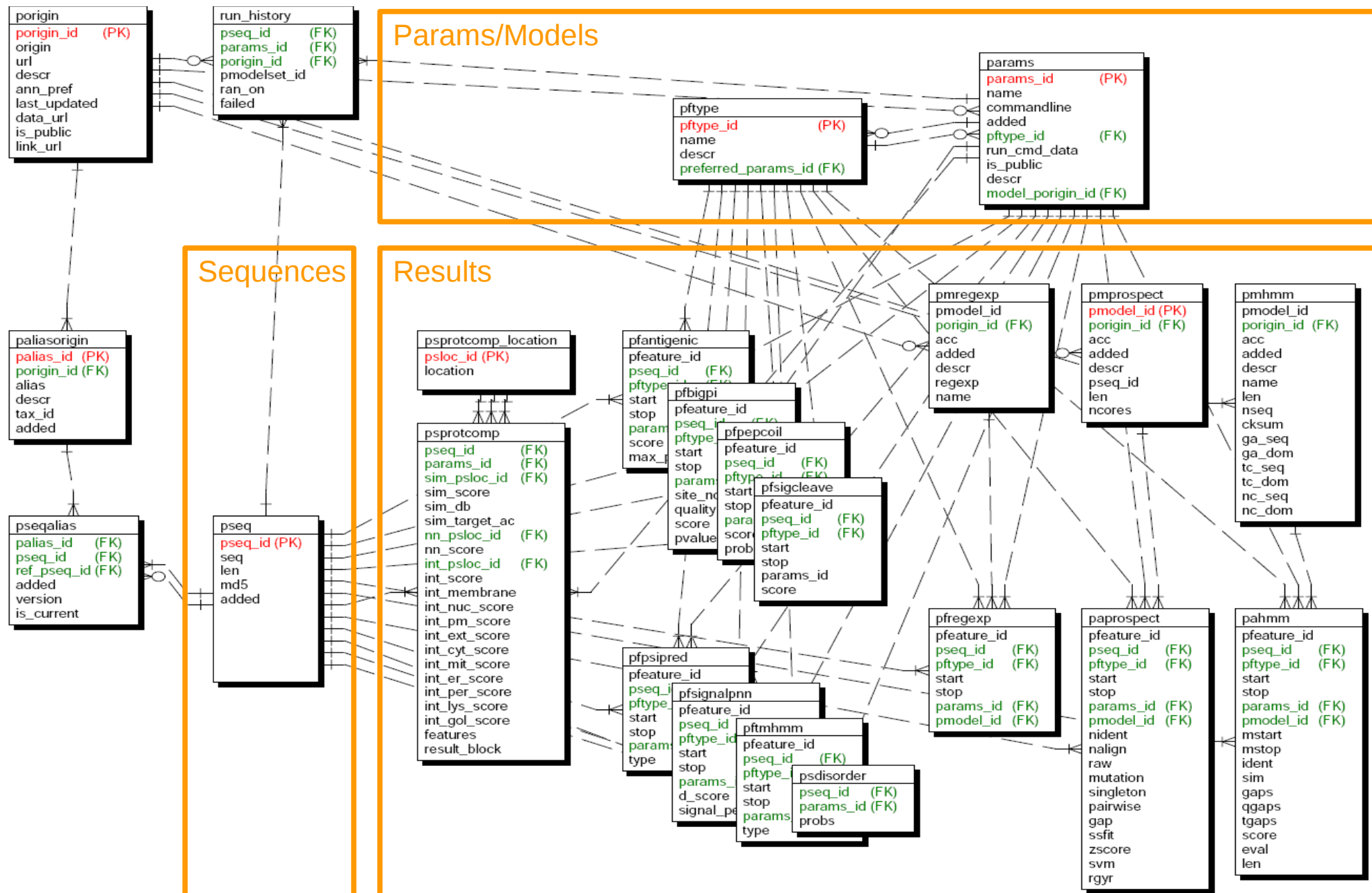
Design Themes

- Abstraction and Normalization
 - most tables are essentially data types
 - expect a lot of joins, but views exist for common queries
 - facilitates updates of new params, etc
- Rely on database for correctness
 - pedantic and paranoid use of triggers and constraints
- Selective incorporation of external databases
 - schemas: unison, ncbi, tax, dali, go, pdb

Results Cube



Schema Overview



Critical Tables, Views, Functions

- Tables and views
 - porigin, pseq, pftype, params
 - palias (view) and current_annotations_v
 - pahmm + pmhmm => pahmm_v
 - pfbigpi => pfbigpi_v
 - pfsignalpnn
 - psprotcomp + psprotcomp_locations => psprotcomp_v
 - pfregex + pmregex => pfregex_v
 - run_history => run_history_v
- Functions
 - pseq_id_from_sequence()
 - params_id() and preferred_params_id_by_pftype()
 - porigin_id()

Table and Column Names

- X_id is always a primary or foreign key
 - foreign and primary keys always have the same name
 - except for pairwise sequence comparisons which use q_pseq_id and t_pseq_id
- psX – protein sequences property
 - e.g., psprotcomp
- pfX – protein feature
 - e.g., pfsignalpnn
- paX and pmX – protein alignments to models
 - e.g., pmhmm and pahmm

View Name Suffixes

- `_mv` – materialized view
- `_dv` – “defining view”
 - mostly for internal use
 - not optimized, often slow
 - look for a corresponding `_mv`
- `_cv` – “canned view”
 - views for complex data mining
 - exposed to public
 - see `canned_views` table
- `_v` – ye olde standard view

Accessing Unison

- How:
 - native protocol
 - *psql interactive shell (akin to sqlplus)*
 - perl DBI
 - *perl API*
 - ODBC/JDBC
 - *web pages (and linking to them)*
 - *command line tools*
 - Oracle snapshot in biodev1 and bioprd1
- Details:
 - host csb
 - database csb or csb-dev
 - as self with Kerberos (user PUBLIC, no password is deprecated)

☞ *Use Kerberos authentication wherever possible. Be prepared to change logins which use PUBLIC.*

Web Page Linking

- http://csb/unison/cgi/pseq_summary.pl?q=<query>
 - query may be pseq_id, alias, md5, or sequence itself
 - all links use GET method (*i.e.*, params in URL)
- Sequence analysis pages
 - summary page
 - aliases
 - patents
 - protein features
 - protein structure and variant mapping
 - HomoloGene homologs
 - pmap and blat genome maps
- Other pages
 - feature-based mining
 - browse “canned views”

unison-get-seq

- Fetch sequences by pseq_id or alias, from stdin or args
- Useful flags:
 - -A select by alias
 - -b “best alias” in define
 - -B “best annotation” in define
 - -u Unison:pseq_id in define (-bu may be used together)
 - -iupac20 select only sequences with standard AA
 - -v verbose – prints progress to stderr
- Example:
 - kinit
 - unison-get-seq -Abu TNFA_HUMAN

```
>Unison:98 UniProtKB/Swiss-Prot:TNFA_HUMAN (Tumor necrosis factor precursor (TNF-alpha)
(Tumor necrosis factor ligand superfamily member 2) (TNF-a) (Cachectin))
MSTESMIRDVELAEEALPKKTGGPQGSRRCLFLSLFSFLIVAGATTLCLLHFGVIGPQR
EEFPRDLSLISPLAQAVRSSSRTPSDKPVAHVVANPQAEGQLQWLNRRANALLANGVELR
DNQLVVPSEGLYLIYSQVLFGQGCPSTHVLLTHTISRIAVSYQTKVNLLSAIKSPCQRE
TPEGAEAKPWYEPIYLGGVFQLEKGDRLSAEINRPDYLDFAESGQVYFGIIAL
```

unison-annotation

- Provides features and other information for sequences specified by pseq_id, alias, or fasta

```
$ unison-annotation -S <myseqs.fa
```

```
*Unison:98:UniProtKB/Swiss-Prot:TNFA_HUMAN (Tumor necrosis fac...
```

```
*signalp (SignalP 3.0 (euk), ran on 2005-11-03 16:40)
```

```
*tm (TMHMM 2.0c, ran on 2005-11-16 14:05)
```

```
#start stop
```

```
35 57
```

```
*regexp (regexp, ran on 2005-11-16 11:26)
```

```
#start stop feature
```

```
133 138 ITIM
```

```
*pfam (Pfam 19.0 ls, ran on 2006-01-13 21:30)
```

```
#start stop score eval acc feature descr
```

```
102 233 210 4.8e-60 PF00229.8 TNF TNF(Tumour Necrosis Factor) family
```

```
*protcomp (protcomp default, ran on 2005-08-10 05:40)
```

```
#loc
```

```
Plasma membrane
```

```
*alia#alias origin descr
```

```
NP_000585 RefSeq tumor necrosis factor alpha [Homo sapiens].
```

```
PR034403 GenenGenes Human TNF-a
```

```
PR021907 GenenGenes Human TNF-a
```

```
PR06 GenenGenes Human TNF-a
```

```
IPI00001671.1 IPI Tumor necrosis factor precursor; SWISS-PROT:P01375|
```

Unison perl API

- Unison perl modeule
 - acts like a subclass of DBI
 - adds many utility methods
 - is stable but homely
- The module is available to all users of `/usr/local/tools/bin/perl` by default
- more info:
 - `perldoc Unison`
 - `/gne/research/apps/unison/examples/`

```
use Unison;
my $u = new Unison;
my $q = $u->pseq_id_by_sequence('YGGFM');
my $np = $u->selectrow_array(<<EOSQL,undef,$q);
    SELECT count(*) FROM patents_v WHERE pseq_id=?
EOSQL
print("$q\t$np\t", $u->best_annotation($q), "\n");
my $sth = $u->prepare('select * from run_history_v where pseq_id=?');
$sth->execute($q);
$sth->dump_results();
```

psql

PostgreSQL's command-line interface

psql intro

- Do this:
 - `kinit`
 - `psql -h csb -d csb`
- The most important commands
 - `\?` – help
 - `\d+ [selection]` – describe object
 - `\dv+ [selection]` – view summary
 - `\dt+ [selection]` – table summary
 - `\df+ [selection]` – function summary
 - selection may be:
 - a table, view, or function name, optionally schema-qualified
 - e.g., `pseq`, `unison.palias`
 - a shell-style glob
 - `explain [query]`

Browsing Tables and Views

```
unison@csb=> \dt+ unison.
```

List of relations				
Schema	Name	Type	Owner	Description
unison	_readme	table	unison	READ THIS FIRST -- Unison
unison	best_annotation_mv	table	unison	materialized view of best
unison	canned_views	table	unison	curated data mining views
unison	ensembl_coordinates_mv	table	unison	
unison	ensembl_unambiguous_coordinates_mv	table	unison	
unison	ensembl_unambiguous_overlaps_mv	table	unison	
unison	genasm	table	unison	genome and assembly

```
unison@csb=> \dv+ unison.*pmap*
```

List of relations				
Schema	Name	Type	Owner	Description
unison	pmap_aln_unambiguous_overlaps_v	view	unison	
unison	pmap_aln_unambiguous_v	view	unison	
unison	pmap_v	view	unison	view of pmap alignments ...

Viewing a Table Definition

```
unison@csb=> \d+ pahmm
```

Table "unison.pahmm"

Column		Description
pfeature_id	//	unique feature id
pseq_id	//	unique protein sequence identifier -- see pseq(pseq_id)
pftype_id	//	protein feature type identifier -- see pftype(pftype_id)
start	//	start of prediction in protein sequence
stop	//	stop of prediction in protein sequence
params_id	//	parameter set identifier -- see params(params_id)
pmodel_id	//	unique protein model identifier
mstart	//	start of match /in model/
mstop	//	stop of match /in model/
ident	//	
sim	//	
gaps	//	
qgaps	//	number of gaps in query sequence
tgaps	//	number of gaps in target sequence
score	//	algorithm-specific score
eval	//	expectation value
len	//	

Indexes:

```
"pahmm_redundant_feature" UNIQUE, btree (pseq_id, "start", stop, pmodel_id,
params_id, mstart, mstop) CLUSTER
```

```
"pahmm_search1" btree (pmodel_id, eval, params_id)
```

```
"pahmm_search2" btree (params_id, eval, pmodel_id)
```

```
"pahmm_search3" btree (params_id, score, pmodel_id)
```

Foreign-key constraints:

```
"pahmm_params_id_exists" FOREIGN KEY (params_id) REFERENCES params(params_id) ON
UPDATE CASCADE ON DELETE CASCADE
```

Viewing a View Definition

```
unison@csb=> \d+ pmap_v
```

View "unison.pmap_v"			
Column	Type	Modifiers	Description
params_id	integer		parameter set identifier -- see params
genasm_id	integer		genome assembly identifier -- see genasm
pseq_id	integer		unique protein sequence identifier -- see pseq
aln_id	integer		pmap_aln alignment identifier
pstart	integer		start of alignment in protein sequence
pstop	integer		stop of alignment in protein sequence
exons	bigint		number of exons
aln_length	bigint		length of alignment
pct_cov	integer		percent coverage
ident	integer		
pct_ident	integer		percent identity
chr	text		chromosome
strand	character(1)		genomic strand ('+' or '-')
gstart	integer		genomic start position on chromosome
gstop	integer		genomic stop position on chromosome

View definition:

```
SELECT a.params_id, a.genasm_id, h.pseq_id, ah.aln_id, min(h.pstart) AS pstart, ...
FROM pmap_hsp h
JOIN pmap_alnhsp ah ON h.hsp_id = ah.hsp_id
JOIN pmap_aln a ON ah.aln_id = a.aln_id
JOIN pseq q ON h.pseq_id = q.pseq_id
GROUP BY a.params_id, a.genasm_id, h.pseq_id, ah.aln_id, h.chr, h.strand, ...
ORDER BY h.pseq_id, (sum(h.pstop - h.pstart + 1)::double precision / ...
```


Reality Queries

(using psql,
but applies to all access modes)

Finding A Sequence

- By sequence
 - `select pseq_id_from_sequence('MYSEQ')`
- By md5
 - `select pseq_id from pseq where md5='f9e8d7...';`
- By alias
 - `select pseq_id from palias where alias='PR054321'`
 - *or* `alias LIKE 'NP_00123%'`
 - *or* `alias ~ '^NP_0123'`
 - `AND porigin_id=porigin_id('RefSeq')`
 - `AND porigin_id=porigin_id('GenenGenes')`
 - `AND tax_id=gs2tax_id('BRARE')`
- By NCBI gene
 - `select pseq_id from palias where porigin_id=porigin_id('RefSeq gi') and alias=73967277;`

Finding Parameters

- select params_id,name,descr from params;

params_id	name	descr	commandline
19	BIG-PI default	BIG-PI GPI prediction; http://	bigpi metazoa %s short
3	BLAST	UNSUPPORTED! Unison internal u	blastall -FF -z2829482 -p blas
36	Bcl-2 ls	Bcl-2 homology domains custom	ldhmpfam --acc -E10 -Z8183
4	EMBOSS/antigenic	antigenicity predictions; http	antigenic -minlen 6 -rformat s
37	EMBOSS/pepcoil	EMBOSS pepcoil coiled-coil pre	pepcoil -noother -window 28 -f
5	EMBOSS/sigcleave	signal cleavage prediction; ht	sigcleave -minweight 3.5 -rfor
11	Genome BLAT	genomic localization of protei	gfClient -t=dnax -q=prot trp 1
38	PMAP 2006-03-20	genomic localization of protei	pmap.2006-03-20 -d NHGD_R35 -B
8	PSSM default		
33	Pfam 19.0 fs	Pfam 19.0 /f/ragmentary models	ldhmpfam --acc -E10 -Z8183
34	Pfam 19.0 ls	Pfam 19.0 local models	ldhmpfam --acc -E10 -Z8183
14	Prospect2 SCOP default	Prospect protein threading; ht	-scop
1	Prospect2 default	Prospect protein threading; ht	-global
2	Prospect2 global_local	Prospect protein threading; ht	-global_local
23	Prospect2 ssp_psiPred default	Prospect protein threading; ht	-global
17	PsiPred v2.45	PSIPRED secondary structure pr	runpsipred -j 3 -h 0.001 -a 2
28	SignalP 3.0 (euk)	Signal sequence prediction per	/gne/compbio/i686-linux-2.6/bi
29	TMHMM 2.0c	TMHMM 2.0c, http://www.cbs.dtu	/gne/compbio/i686-linux-2.6/op
41	dispro	http://www.ics.uci.edu/~baldig	MANUAL
39	disprot VL3H	disprot protein disorder predi	MANUAL: temple-disprot.pl
31	gne_prospect	Prospect protein threading; ht	gne_prospect -a 2 -p -b
0	no args		
20	protcomp default	ProtComp protein localization	protcomp
12	regexp		
9	tmDetect default	Genentech in-house TM detectio	tmDetect

- commandline in params
- Consider using:
 - select params_id('Pfam 19.0 ls')
 - select preferred_params_id_by_pftype('HMM')

What's Been Run?

- **Always** consult `run_history_v` to see whether Unison contains the results you seek.

```
unison@csb=> select * from run_history_v where pseq_id=76;
```

pseq_id	params_id	params	//	ran_on	failed
76	19	BIG-PI default	//	2005-02-06 01:50	f
76	3	BLAST	//	2003-08-18 14:36	f
76	36	Bcl-2 ls	//	2006-02-01 19:50	f
76	4	EMBOSS/antigenic	//	2003-10-07 20:56	f
76	37	EMBOSS/pepcoil	//	2006-05-03 18:37	f
76	5	EMBOSS/sigcleave	//	2003-10-13 16:30	f
76	11	Genome BLAT	//	2004-02-11 10:48	f
76	38	PMAP 2006-03-20	//	2006-03-23 14:53	f
76	33	Pfam 19.0 fs	//	2006-01-12 03:05	f
76	34	Pfam 19.0 ls	//	2006-01-13 21:20	f
76	1	Prospect2 default	//	2004-02-25 15:14	f
76	1	Prospect2 default	//	2004-02-20 08:03	f
76	17	Psipred v2.45	//	2005-06-11 07:51	f
76	28	SignalP 3.0 (euk)	//	2005-11-03 16:22	f
76	29	TMHMM 2.0c	//	2005-11-16 08:34	f
76	41	dispro	//	2006-05-04 22:42	f
76	39	disprot VL3H	//	2006-05-04 17:42	f
76	20	protcomp default	//	2005-08-10 03:11	f
76	12	regex	//	2005-11-16 11:26	f
76	12	regex	//	2006-04-13 04:47	f
76	9	tmdetect default	//	2006-02-21 17:38	f

Fetching Pfam Domains

- The hard way

```
unison@csb-dev=> SELECT A.pseq_id,M.name,M.acc,A.eval,A.score,M.descr
FROM pahmm A
JOIN pmhmm M on A.pmodel_id=M.pmodel_id
WHERE A.params_id=34 AND A.eval<1e-10 AND pseq_id=15;
```

pseq_id	name	acc	eval	score	descr
15	Insulin	PF00049.8	6e-23	86	Insulin/IGF/Relaxin family
15	IGF2_C	PF08365.1	2.2e-35	128	Insulin-like growth factor II E-

(2 rows)

Time: 1.900 ms

Fetching Pfam Domains 2

- A better way
 - use the pahmm_v view
 - for queries you want to rerun, use preferred_params_id_by_pftype

```
unison@csb-dev=> SELECT pseq_id,name,acc,eval,score,descr
FROM pahmm_v
WHERE params_id=preferred_params_id_by_pftype('HMM') AND eval<1e-10 AND pseq_id=15;
```

pseq_id	name	acc	eval	score	ends	descr
15	IGF2_C	PF08365.1	2.2e-35	128	[]	Insulin-like growth factor...
15	Insulin	PF00049.8	6e-23	86	[]	Insulin/IGF/Relaxin family...

(2 rows)

Time: 2.574 ms

Fetching “All” Features

- pseq_features_v provides *current* results for *common* feature

```
unison@csb-dev=> SELECT feature,start,stop,score,eval,descr FROM pseq_features_v WHERE  
pseq_id=82;
```

feature	start	stop	score	eval	descr
SS	1	31	0.884		signal sequence
TM	7	29			transmembrane domain
PAN_1	34	124	64	3.9e-16	PAN domain
Kringle	128	206	178	2e-50	Kringle domain
Kringle	211	288	178	1.6e-50	Kringle domain
Kringle	305	383	181	2e-51	Kringle domain
Kringle	391	469	155	1.3e-43	Kringle domain
Trypsin	495	716	223	4.3e-64	Trypsin
Y-ITIM	619	628			ITIM motif with second upstream Y; more ...
ITIM	623	628			Immunotyrosine Inhibition Motif

(10 rows)

Time: 8.624 ms

Wrap Up

Dos and Don'ts

(Not “Dos and Donuts”)

- Do

- specify params_id in all queries for results.
- consult the run_history_v view, especially when your query returns no results.
- use the helper functions params_id, params_id_by_pftype, porigin_id, etc.
- experiment with Unison.
- provide feedback and suggestions.

- Don't

- hesitate to ask questions.
- rely on PUBLIC user.

Developer Resources

- Unison
 - <http://csb/unison/tour/>
 - [/gne/research/apps/unison/examples/](http://gne/research/apps/unison/examples/)
 - `perldoc Unison`
- PostgreSQL
 - `psql \d` commands
 - <http://www.postgresql.org/docs/>
- `cvs -d :ext:csb/srv/cvs unison`
 - contains code examples
- `cvs -d :ext:csb/srv/cvs unison-web`

The Future

- Kiran will assume a greater role for Unison
 - nearly all data loading and maintenance
 - user help
 - API maintenance
 - Oracle data pushes
- I will continue to be involved in
 - mining efforts using Unison
 - added new data types
 - schema, API, and web oversight