FlyMine

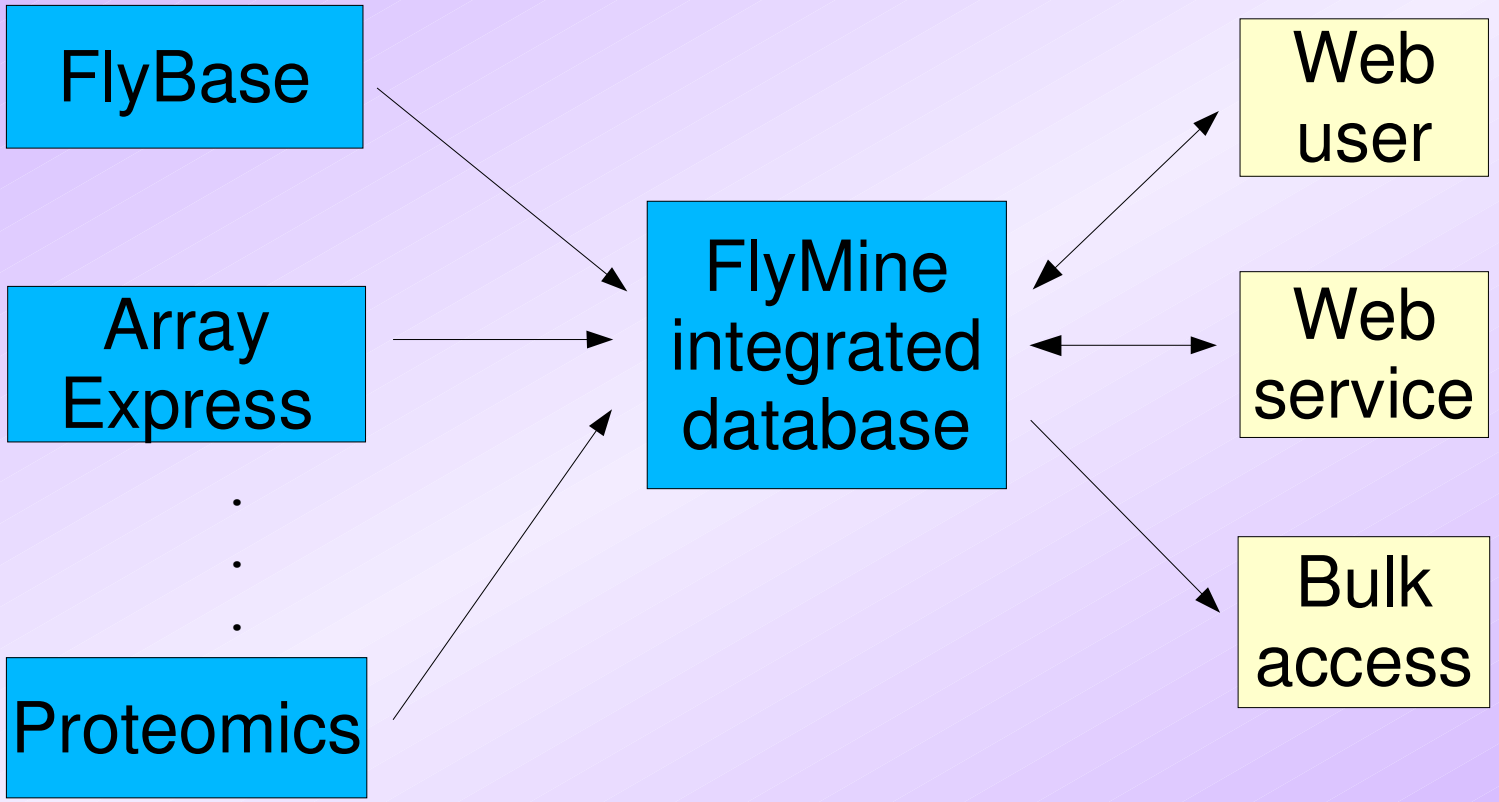
An integrated database for *Drosophila* and *Anopheles* genomics

Andrew Varley
24th June 2003

Why yet another database?

- Currently lots of little databases
  - fine for “browsing”, bad for “querying”
- Hard to query across them
- Massive amounts of experimental data (microarray, proteomics) being produced
- Need to tie this information together
- Chado etc. did not exist
  - FlyMine has broader scope

Outline

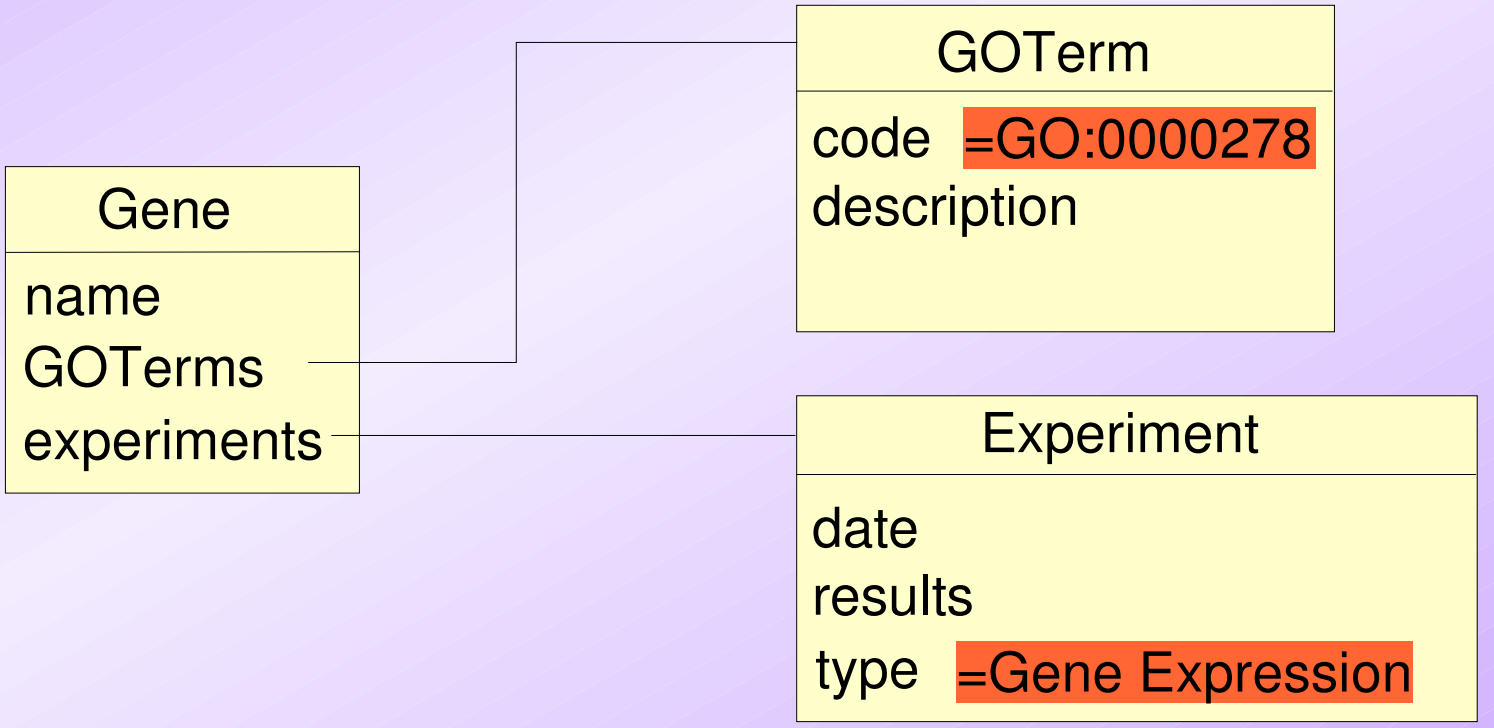

FlyBase
Array Express
Proteomics
FlyMine integrated database
Web user
Web service
Bulk access

Aims

- Integrate data from multiple sources
- Allow arbitrary queries from users
- Queries based on objects, not SQL
- Multiple query interfaces (Java, OQL, graphical, etc)
- Different classes of user (web-based, SOAP)
- Open source!

Query interfaces – graphical

"Show gene expression data for genes which have GO term GO:0000278 applied"



Query interfaces – OQL

"Show gene expression data for genes which have GO term GO:0000278 applied"

```
SELECT gene, exp
FROM Gene AS gene, GOTerm AS term,
     Experiment AS exp
WHERE gene.GOTerms CONTAINS term
AND term.code = "GO:0000278"
AND gene.experiments CONTAINS exp
AND exp.type = "Gene Expression"
```

Query interfaces – Java

"Show gene expression data for genes which have GO term GO:0000278 applied"

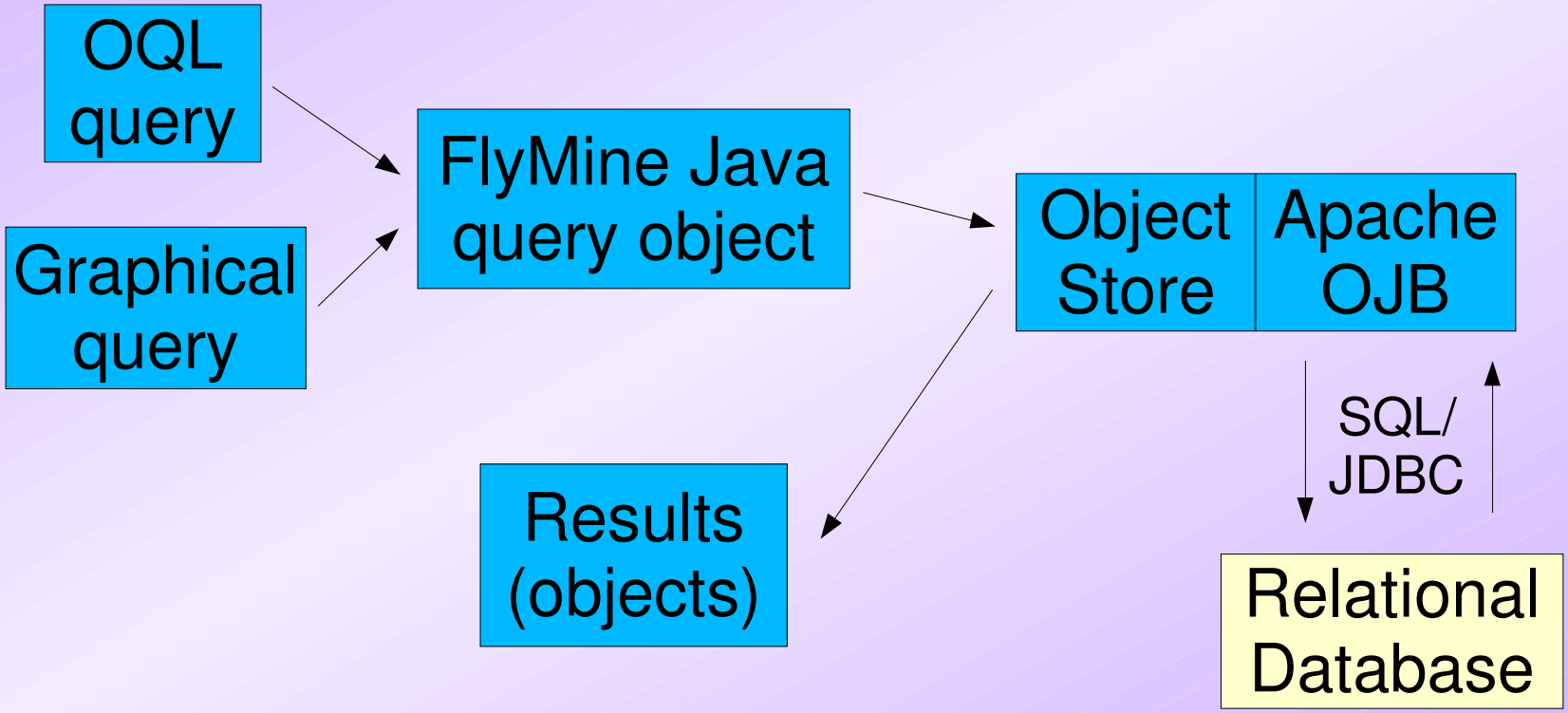
```
Query q = new Query();
QueryClass qcGene = new QueryClass(Gene.class);
QueryClass qcTerm = new QueryClass(GOTerm.class);
QueryField qfGeneTerm = new QueryField(qcGene, "GOTerms")
ConstraintSet cs = new ConstraintSet();
Constraint con1 = new ContainsConstraint(qfGeneTerm,
                  ContainsConstraint.CONTAINS, qcTerm);
cs.add(con1);
.
.
q.setConstraint(cs);
```

Object databases

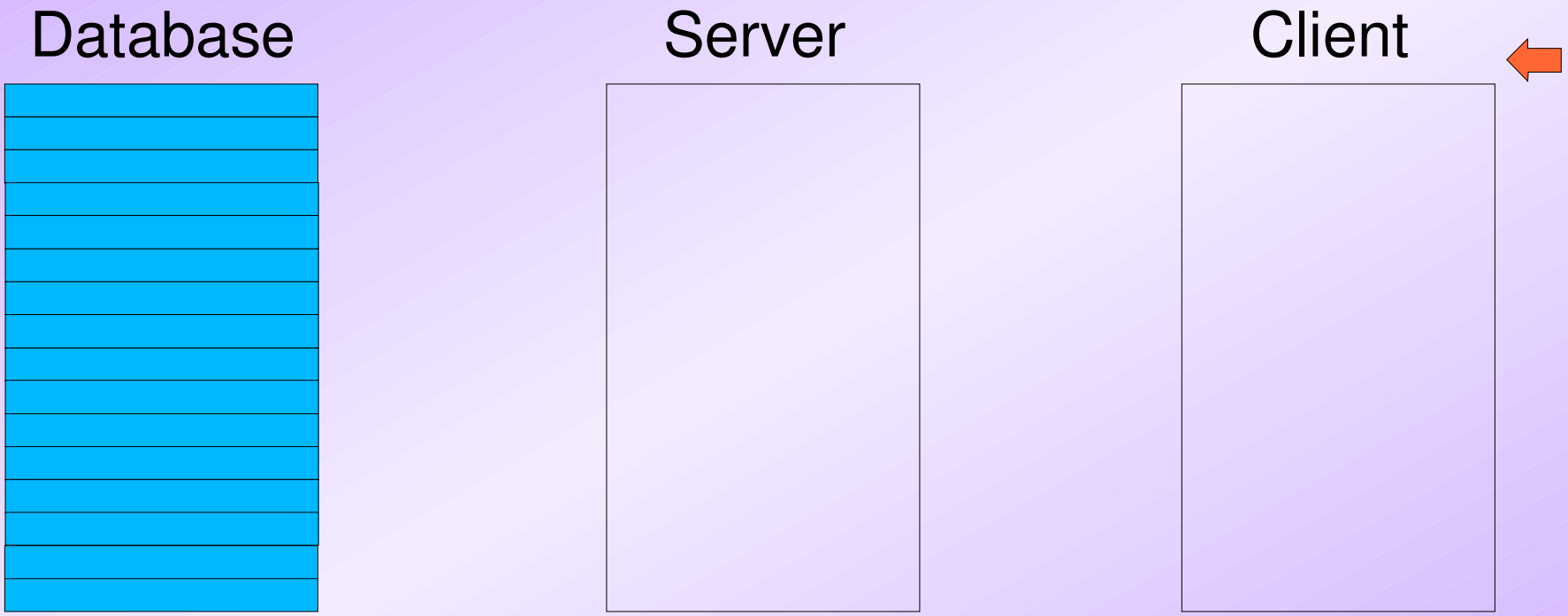
- Open-source solutions include
  - Apache OJB
  - Hibernate
  - Castor
  - various object databases
- Commercial
  - various JDO vendors

FlyMine ObjectStore

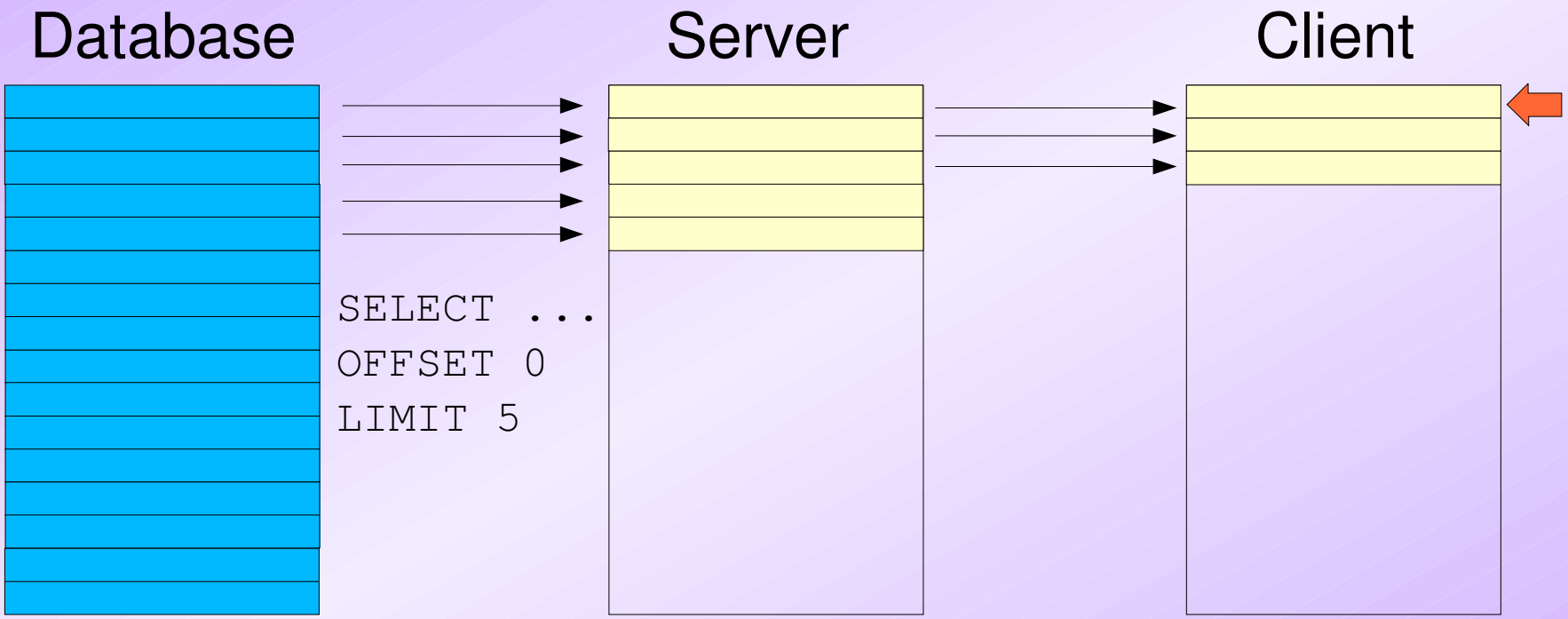
- Based on heavily modified Apache OJB



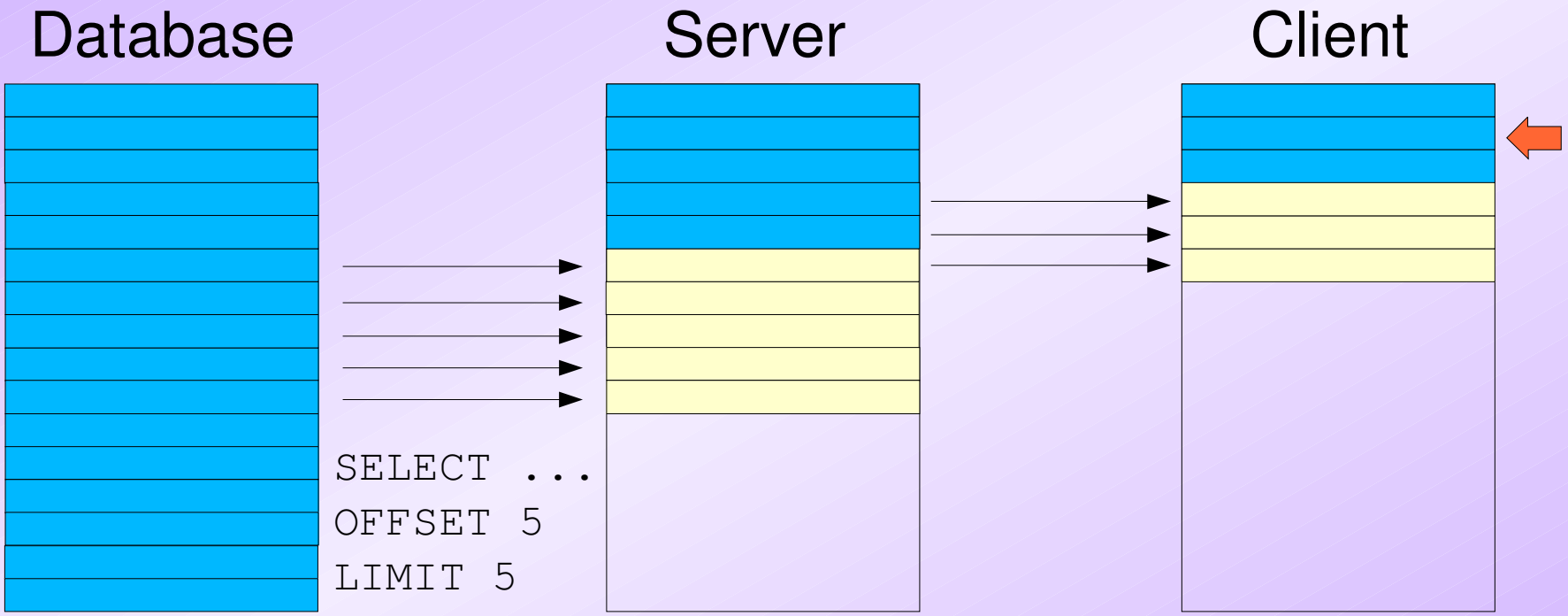
Results



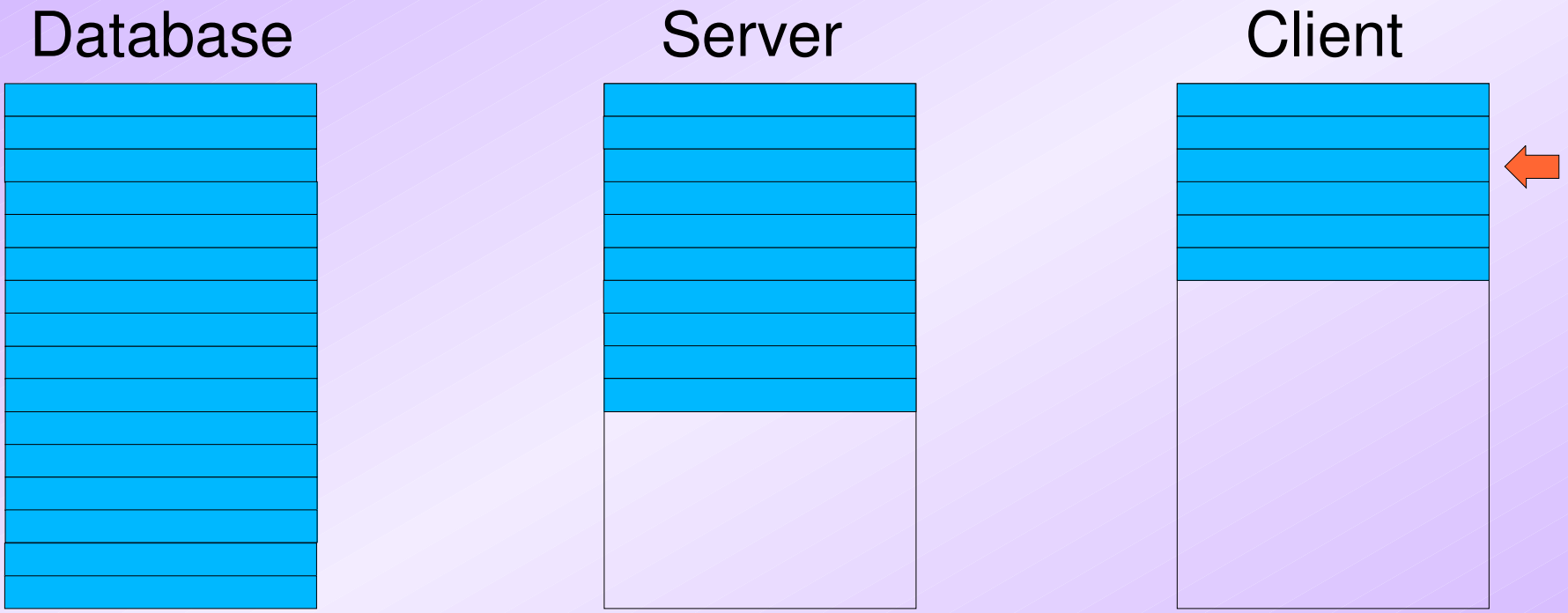
Results



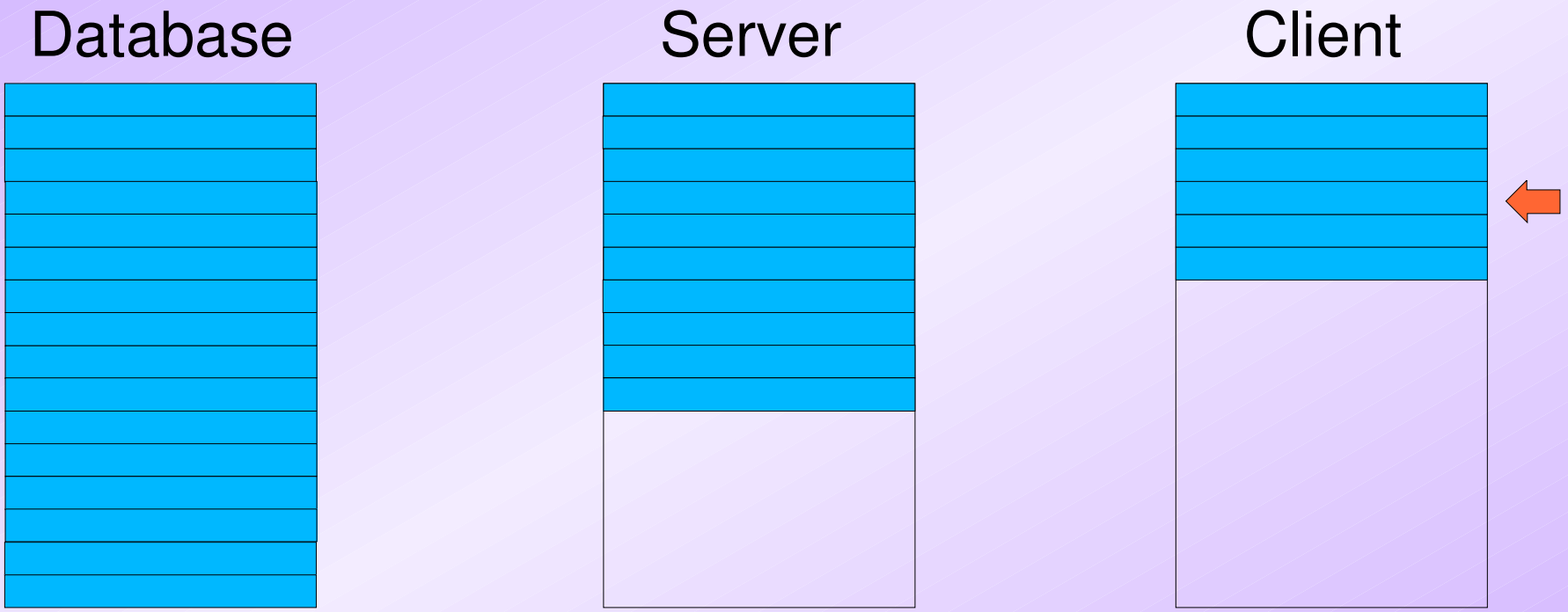
Results



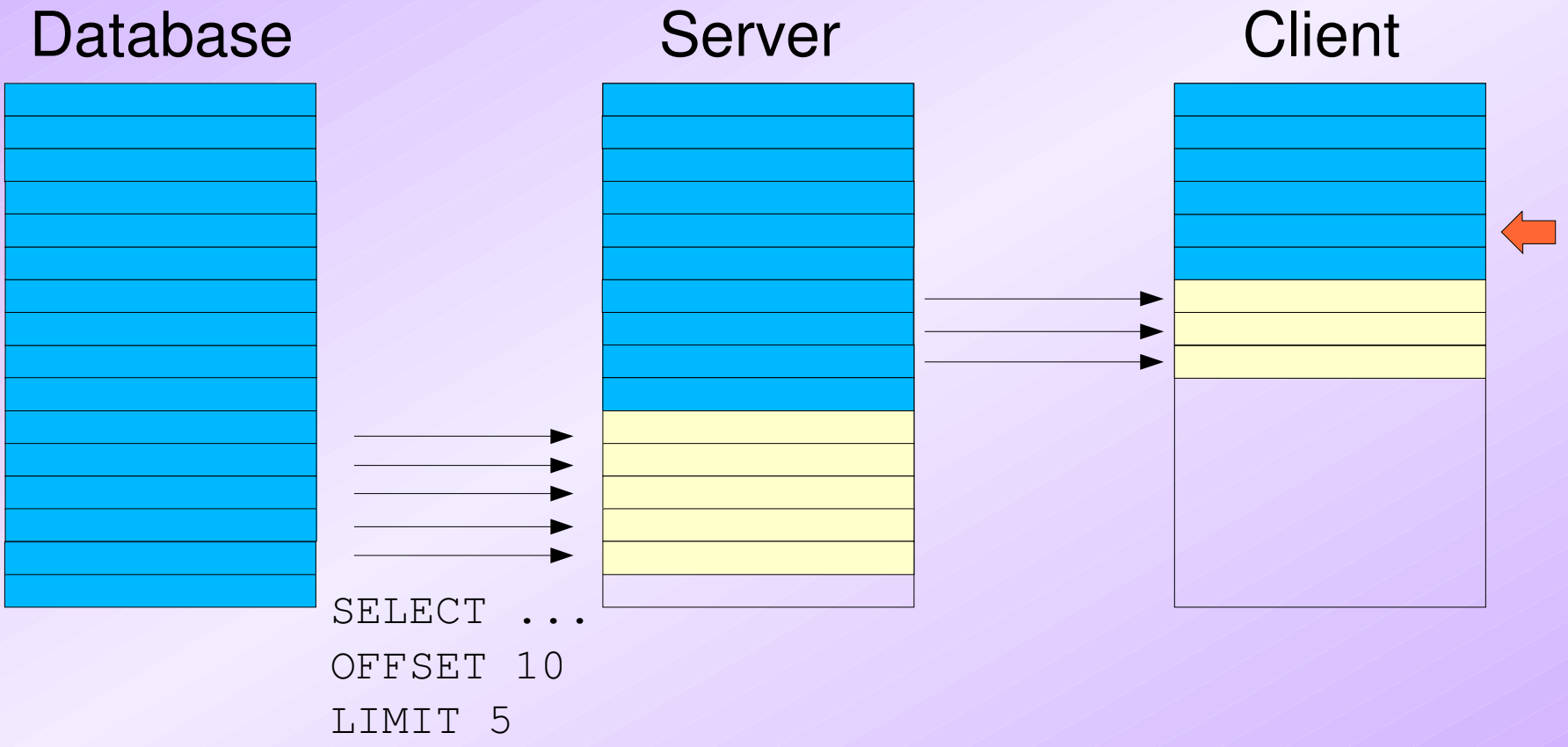
Results



Results

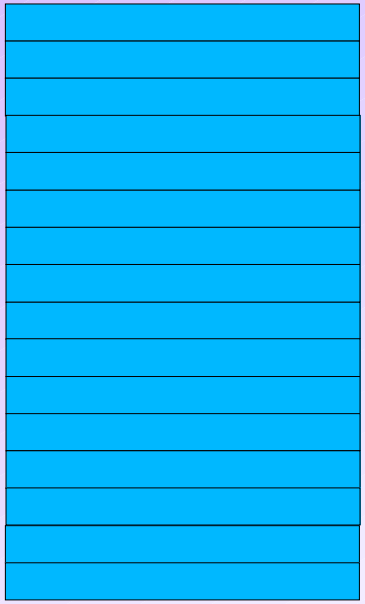


Results

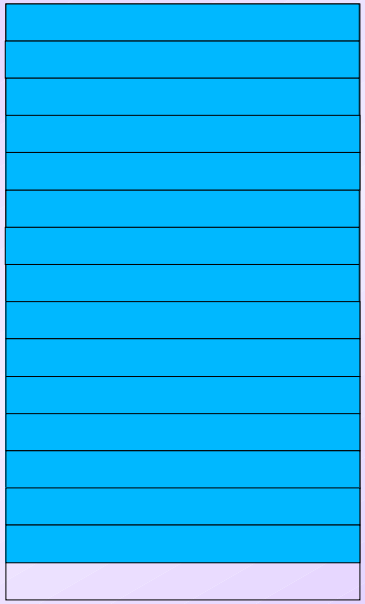


Results

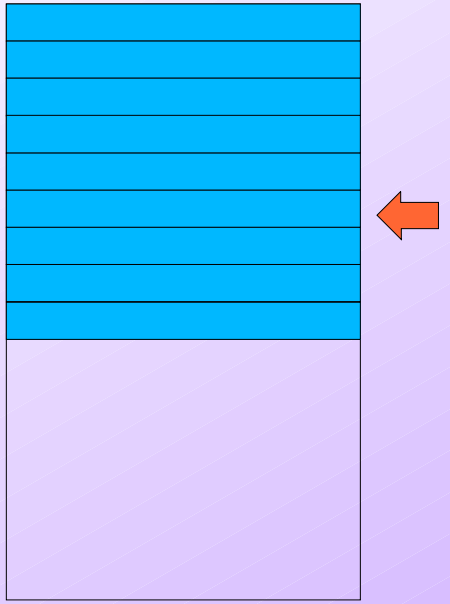
Database



Server

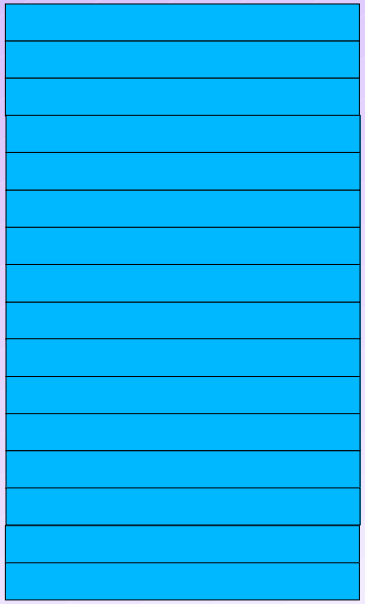


Client

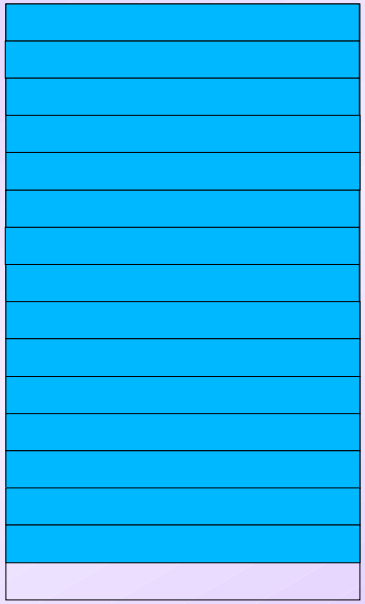


Results

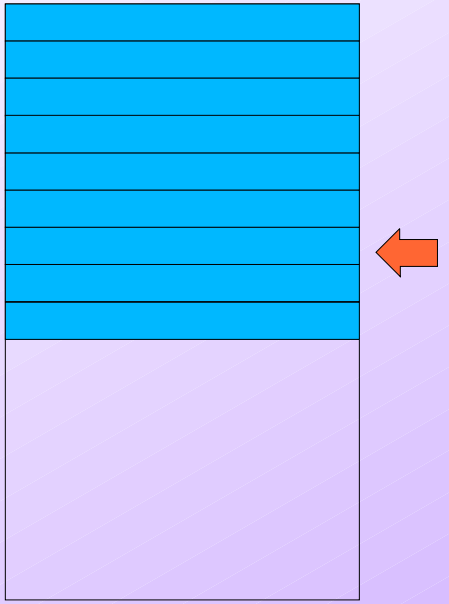
Database



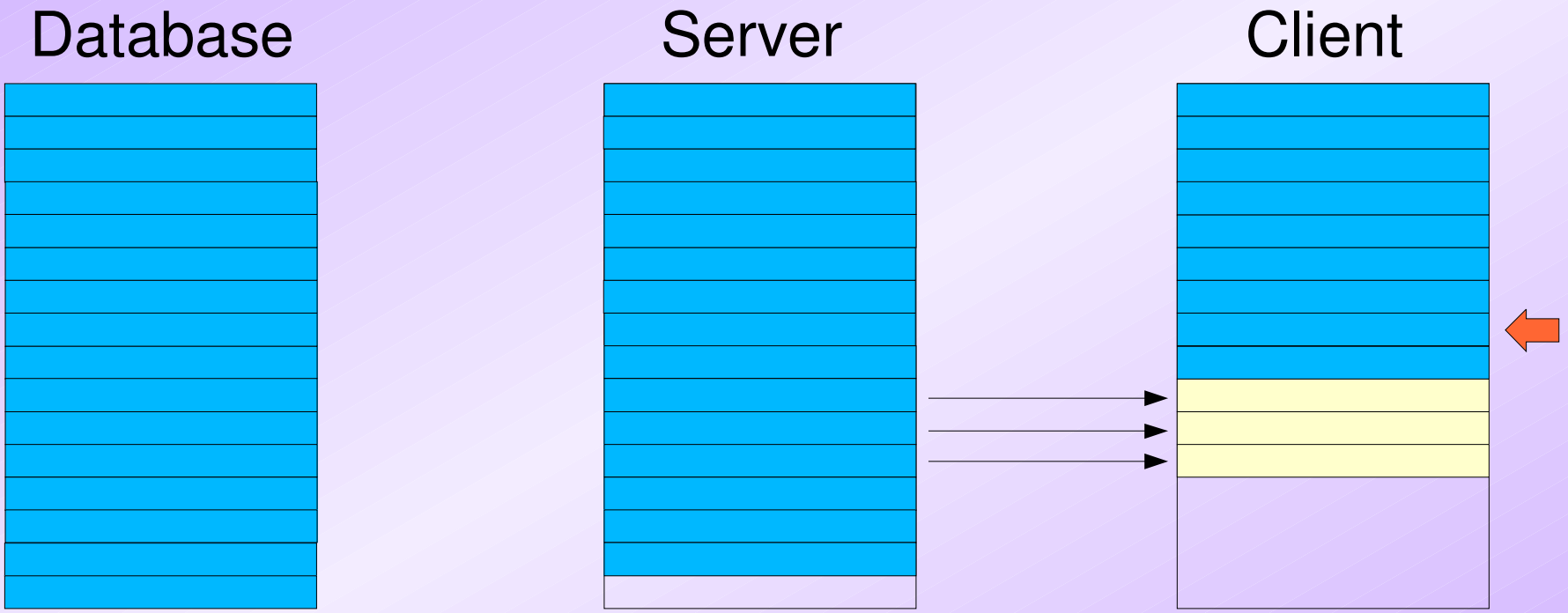
Server



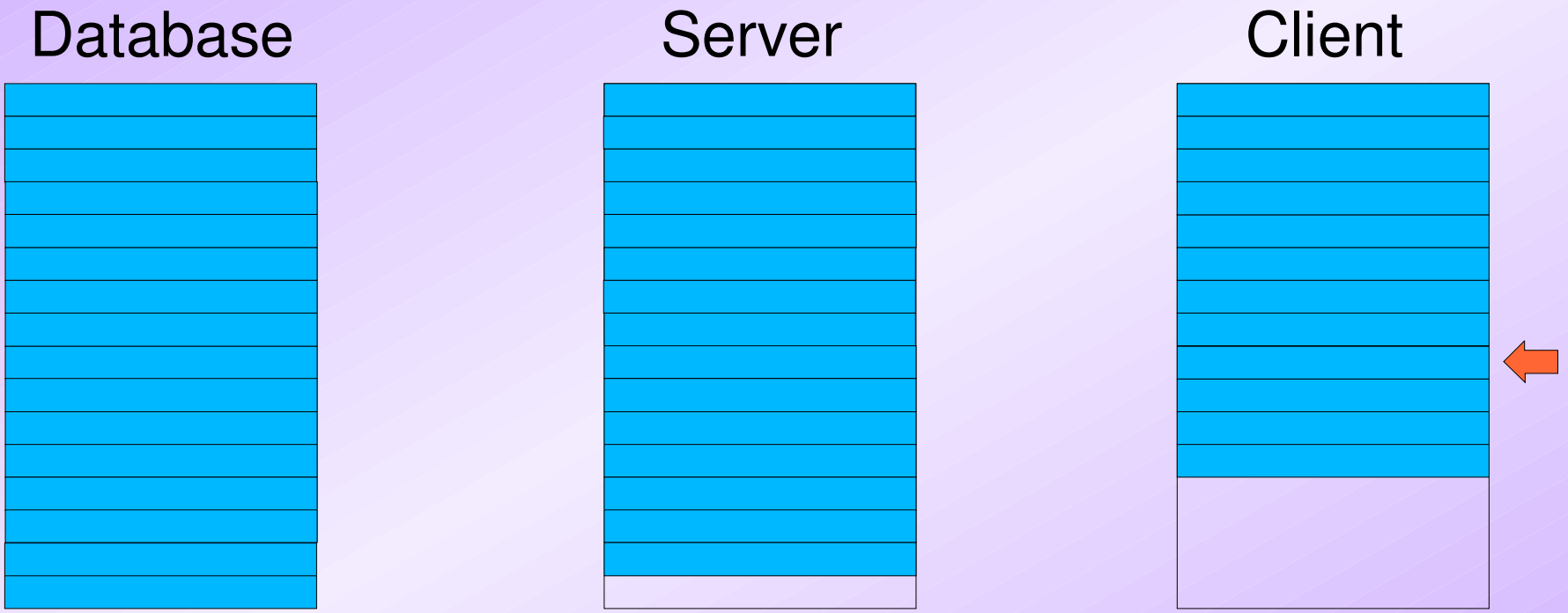
Client



Results

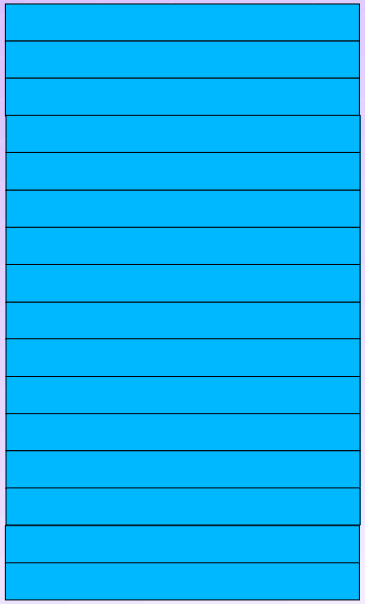


Results

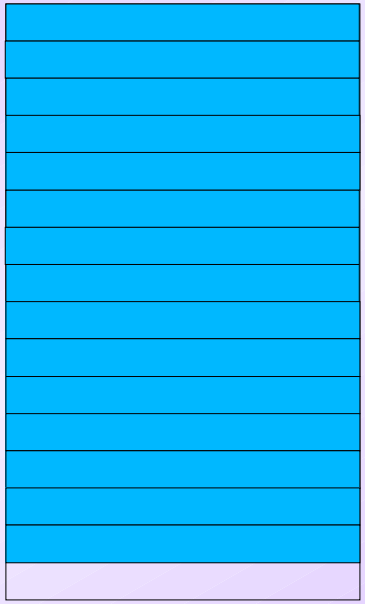


Results

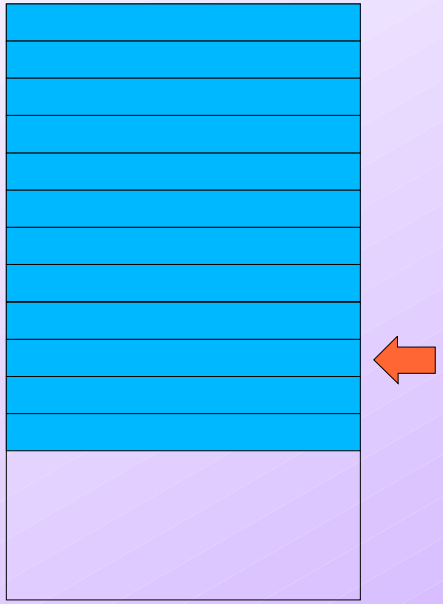
Database



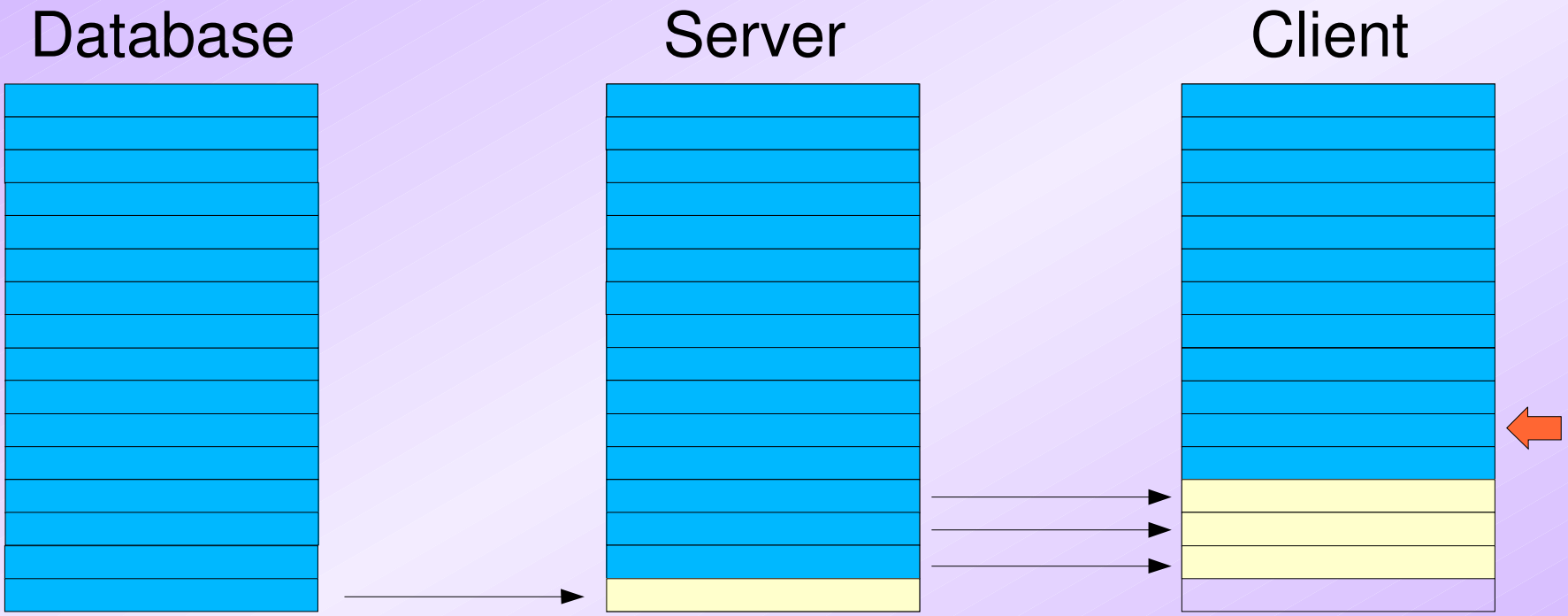
Server



Client



Results




```
SELECT ...
OFFSET 15
LIMIT 5
```

Arbitrary queries – problems

- Badly formed queries may overload database server
- Difficult to optimise database for all queries
  - Which indexes to use?
  - Slow response to complex queries involving multi-table joins

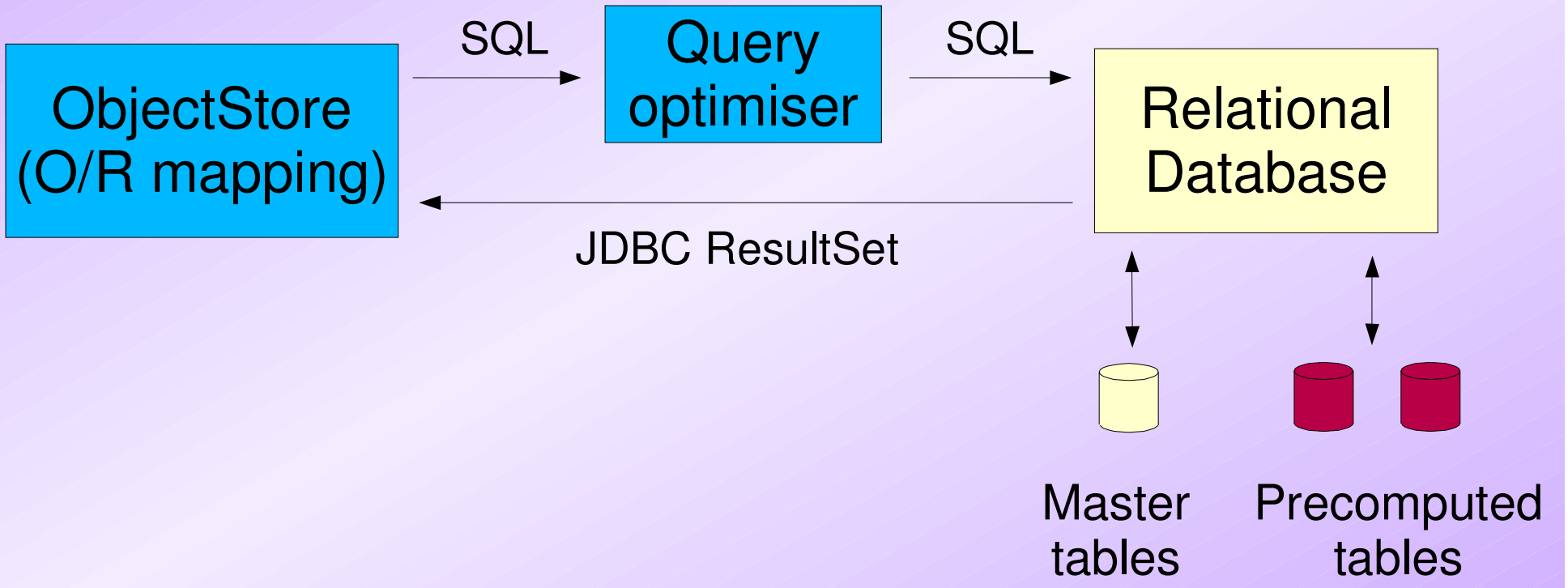
Arbitrary queries – solutions

- Close relationship with database server query planner
  - ask how long a query will take before attempting to run it (~3ms)
  - Disallow queries that will take longer than a certain threshold
- Store data massively redundantly in “precomputed tables” and rewrite incoming queries on-the-fly.

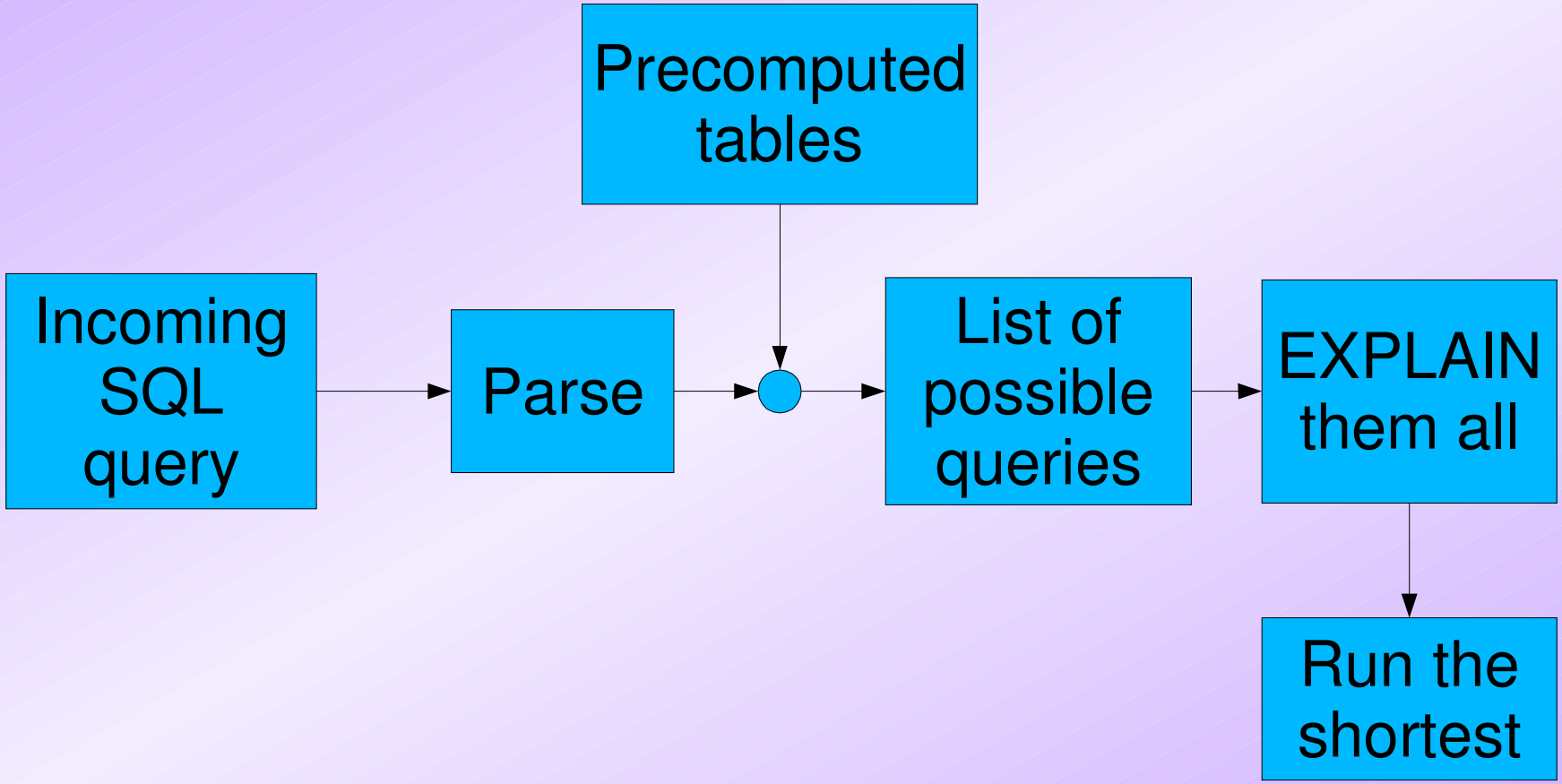
Query optimisation – aims

- Provide a generic way of speeding up complex queries to any read-only SQL database
- Minimal parsing/computational overhead
- Transparent to users/applications
  - no new schema/model to learn
- Make available as standalone module

Query optimisation architecture



Query optimisation



Query optimisation – example

Master tables

Genes

id	name

GOTerms

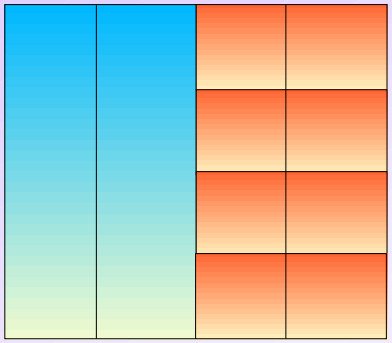
geneid	code

Experiments

geneid	date	type

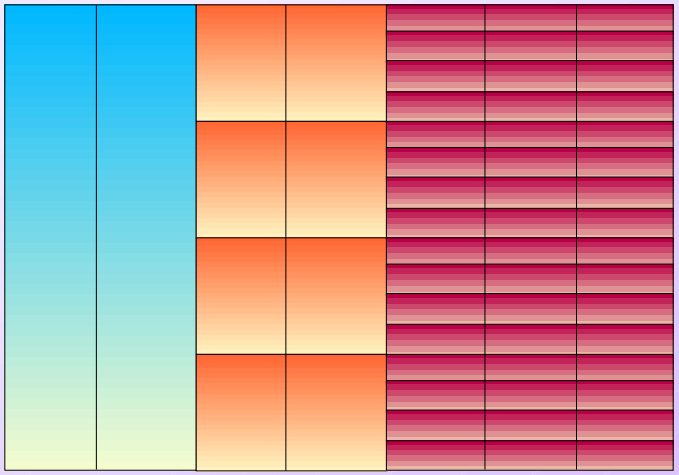
Precomputed tables

GenesGOTerms



```
CREATE TABLE genesgoterms AS
SELECT genes.id AS genes_id,
       genes.name AS genes_name,
       goterms.geneid AS goterms_geneid,
       goterms.code AS goterms_code
FROM genes, goterms
WHERE genes.id = goterms.geneid
```

GenesGOTermsExperiments



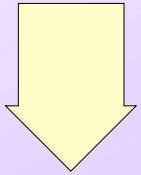
```
CREATE TABLE ....
SELECT ....
FROM genes, goterms, experiments
WHERE genes.id = goterms.geneid
AND genes.id = experiments.geneid
```

FlyMine

Query optimisation – example

"Show dates that gene expression experiments were performed on genes which have GO term GO:0000278 applied"

```
SELECT genes.name, experiments.date
FROM genes, goterms, experiments
WHERE genes.id = goterms.geneid
AND genes.id = experiments.geneid
AND goterms.code = "GO:0000278"
AND experiments.type = "Gene Expression"
```

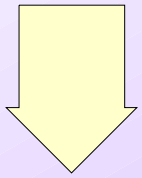


EXPLAIN

5 seconds

Query optimisation – example

```
SELECT genes_name AS name,
       experiments.date
FROM genesgoterms, experiments
WHERE genes_id = experiments.geneid
AND goterms_code = "GO:0000278"
AND experiments.type = "Gene Expression"
```

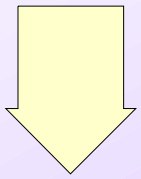


EXPLAIN

1 second

Query optimisation – example

```
SELECT genes_name AS name,
       experiments_date AS date
FROM genesgotermsexperiments
WHERE goterms_code = "GO:0000278"
AND experiments_type = "Gene Expression"
```



EXPLAIN

200ms

Query optimisation – example

- Without optimiser

  Time = 3ms + 5 seconds

  (EXPLAIN) (EXECUTE)

- With optimiser

  Time = 9ms + 100ms + 200ms

  (3 EXPLAINs) (PARSE) (EXECUTE)

Query optimisation – summary

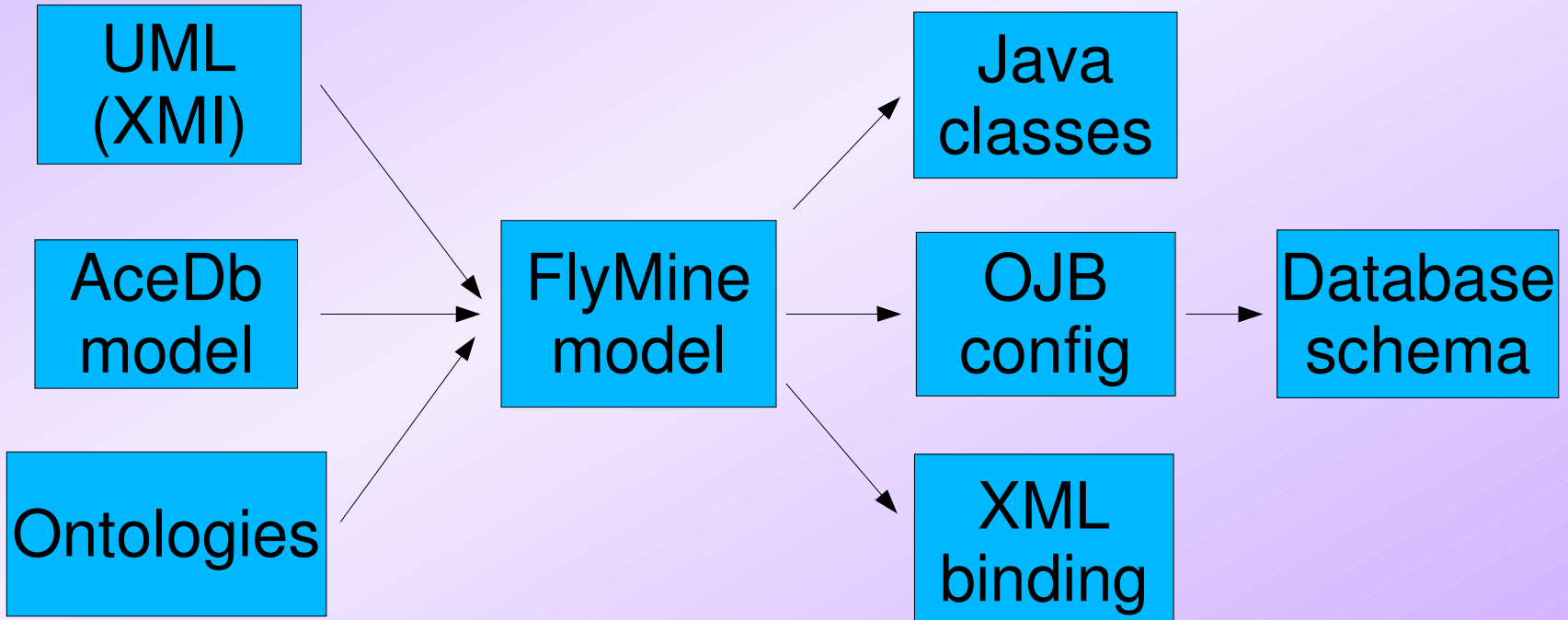
- For complex queries, our SQL optimisation module can produce large speed increases
- Optimisation is transparent to the user or application generating the SQL.
- Choosing which precomputed tables to store is important – may need to analyse incoming queries.

Query optimiser – summary

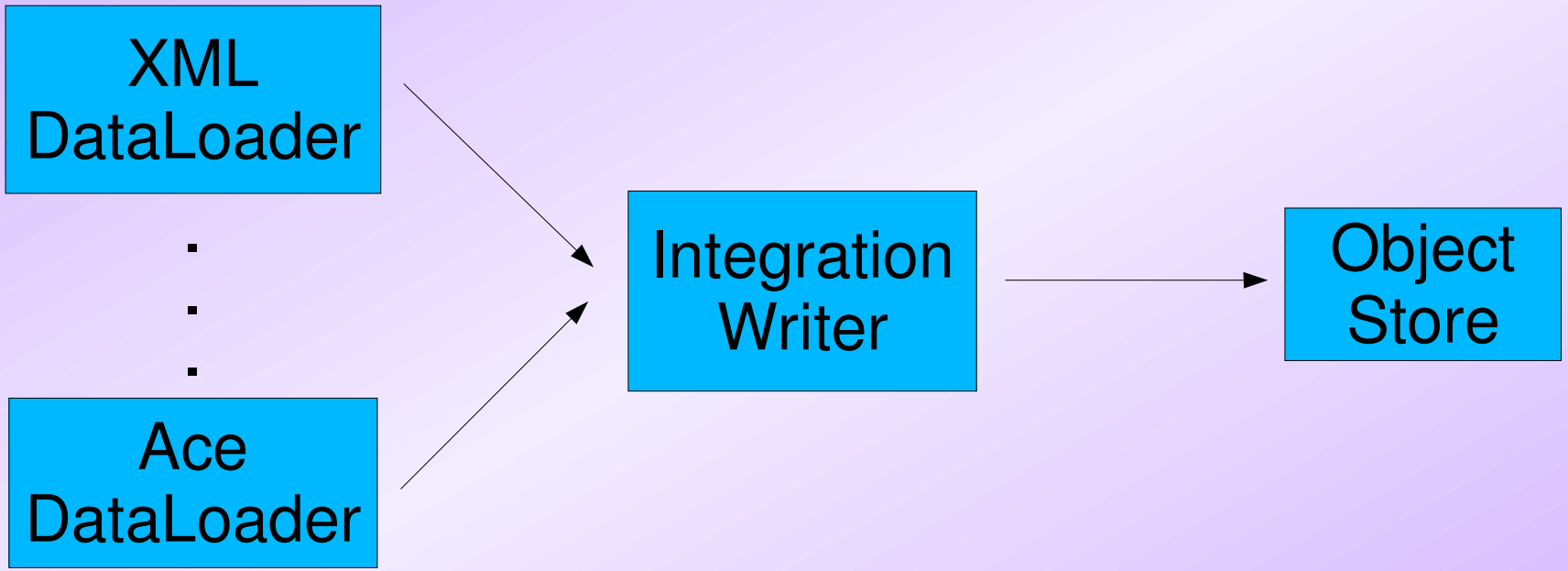
- Optimiser trades off disk space for improved query performance.
- Schema independent.
- Can be used in conjunction with P6Spy to intercept JDBC calls from existing software to a database.
- Available from www.flymine.org !!

Data model

- Data model can be generated from one of a number of sources



Data loading and integration



Summary

- Current status
  - generic SQL query optimiser
  - powerful object database
  - 2 query interfaces (OQL + Java)
  - framework for data loading/integration
  - autogeneration of model-dependent parts from UML or Ace model

Summary

- Coming up
  - graphical query interface
  - web front end
  - web service
  - integrated Drosophila/Anopheles database (genomics, microarray, proteomics, etc)
  - tools for auto-generation of best set of precomputed tables

Acknowledgements

The FlyMine team:

Andrew Varley
Richard Smith
Matthew Wakeling
Mark Woodbridge
François Guillier
Rachel Lyne
Rajasekhar Paidi
Gos Micklem

Download FlyMine from www.flymine.org

FlyMine is funded by the Wellcome Trust (grant no. 067205), awarded to M. Ashburner, G. Micklem, S. Russell, K. Lilley and K. Mizuguchi.