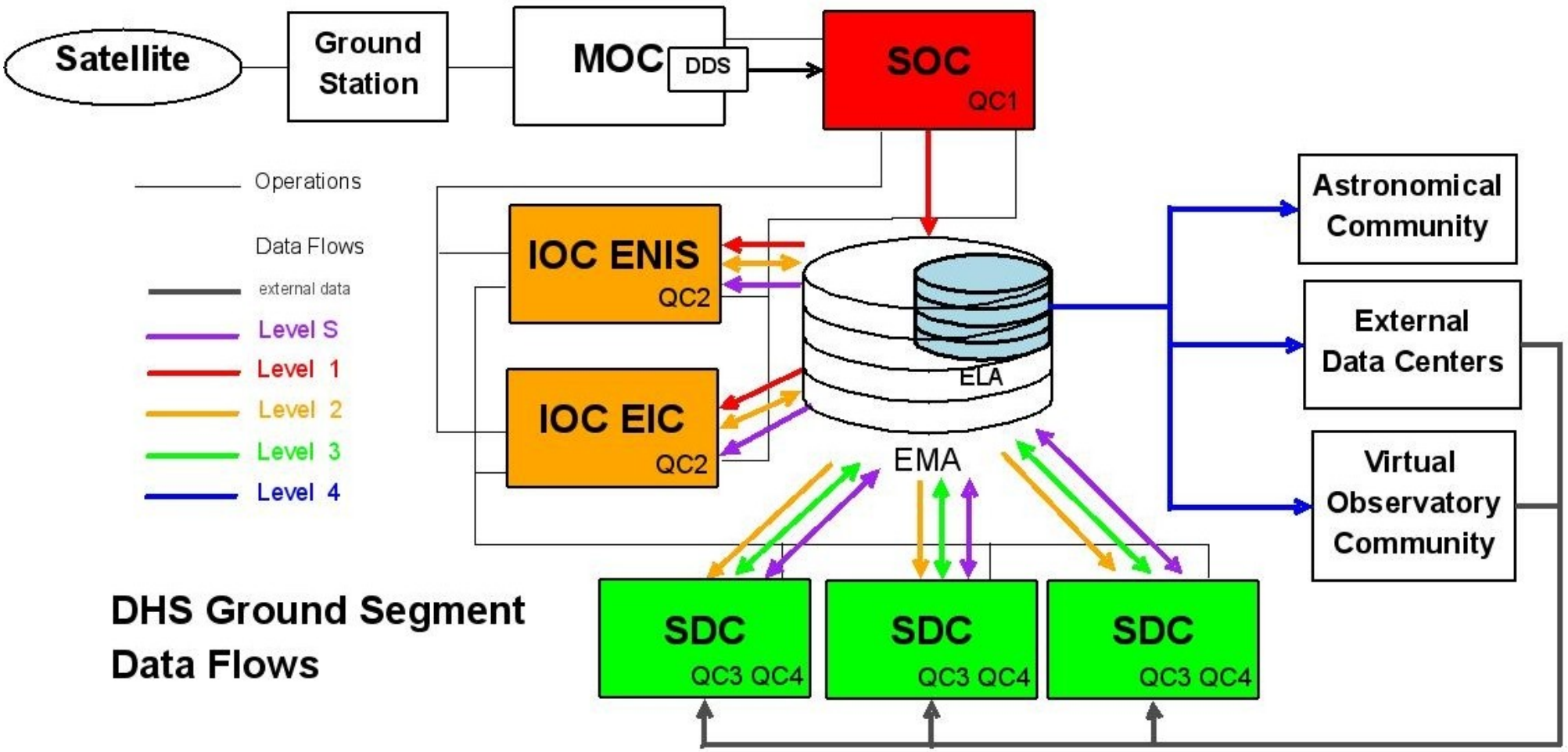
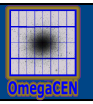
Virtual Observations 2016
Data Mining in Astronomy

RDBMS & SQL

Data-centric approarch



Database

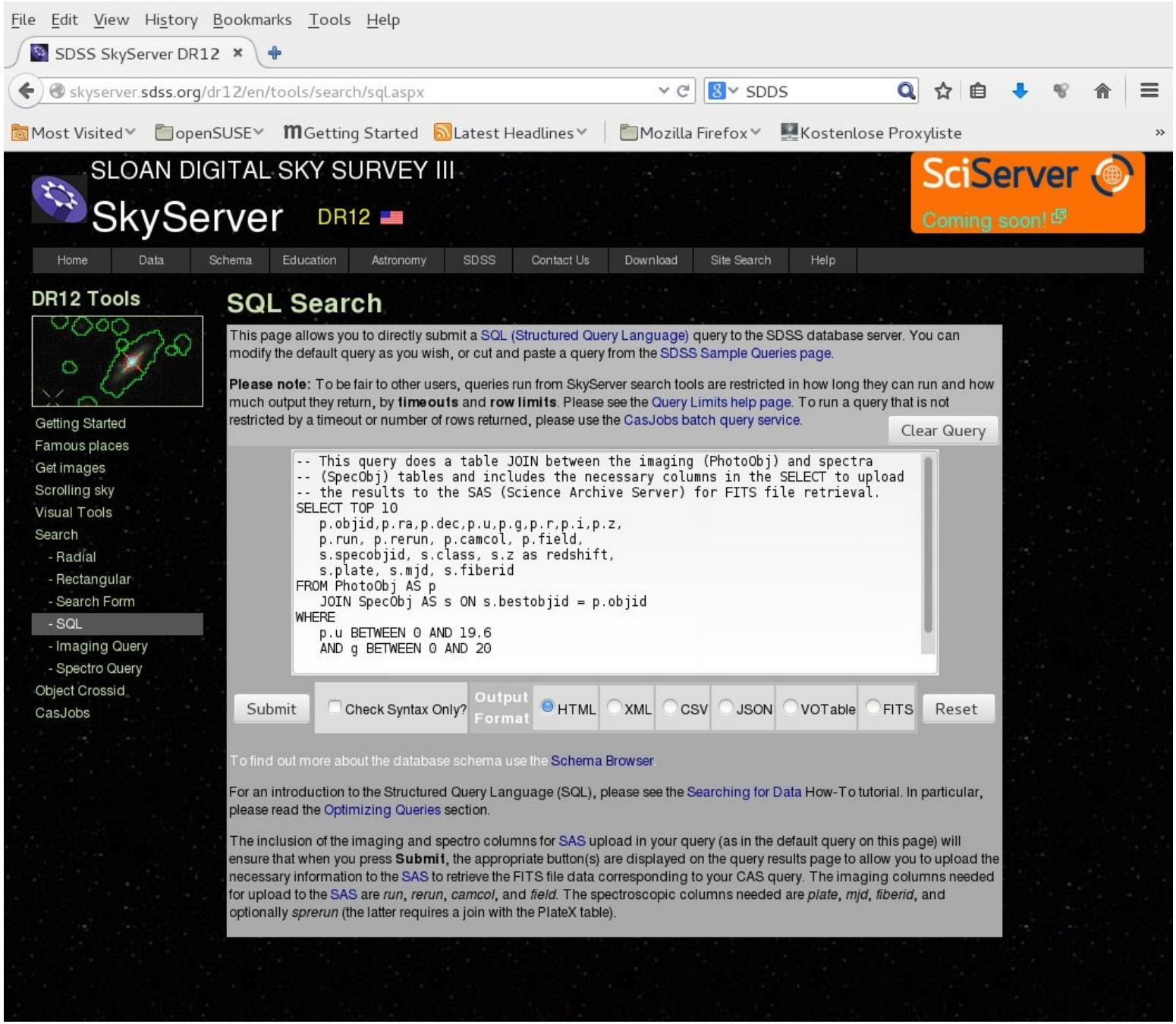


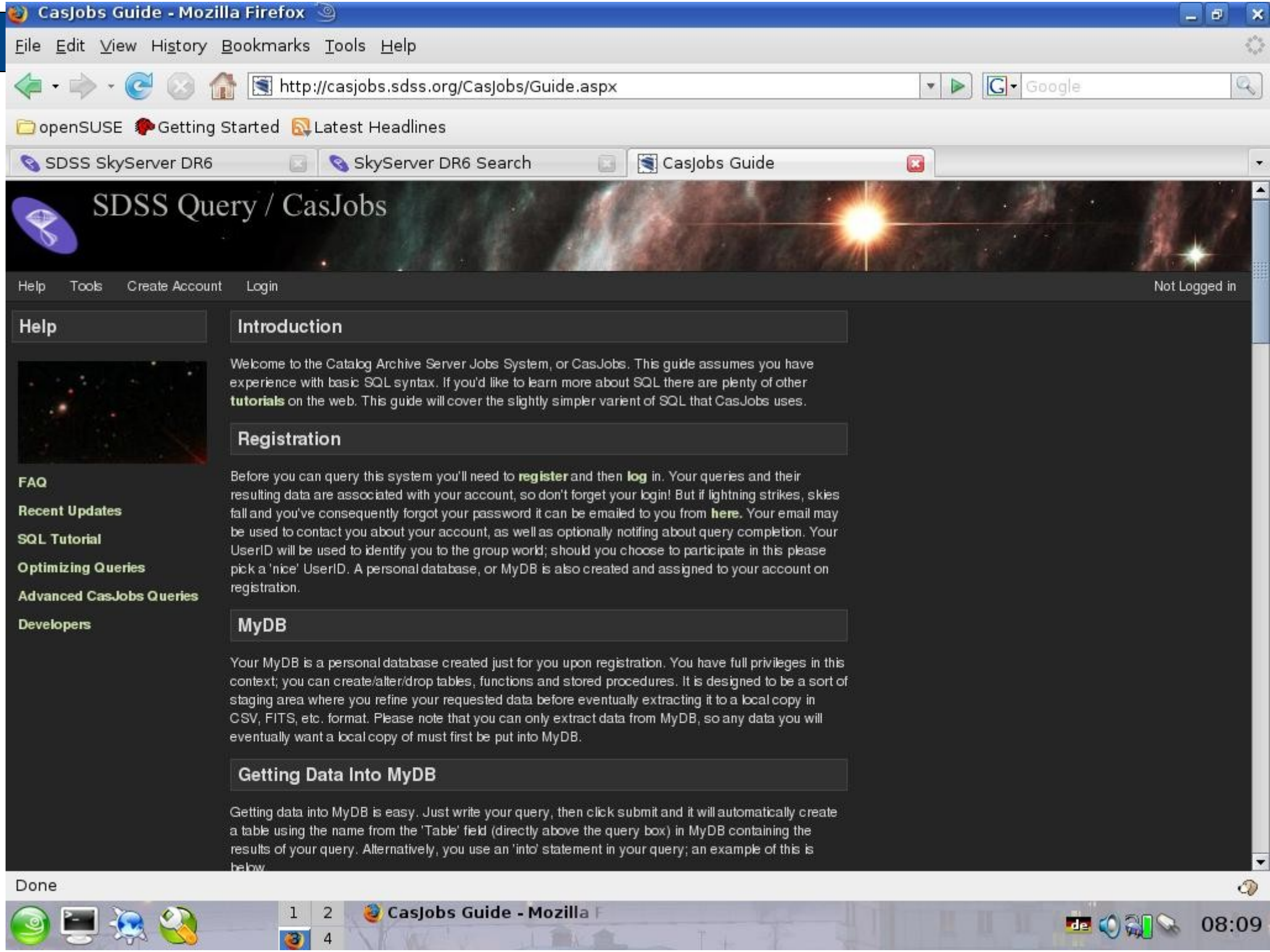
- Definition of database
- Relational and Object-oriented databases
- Relational algebra
- SQL
- Normalization
- Joins, substatements etc.

Database – an example

- 2,8 mln objects
- Coordinates, Ids, pm, spectral information, photometry, bibliography
- Most citated astronomical DB up to date
- http://simbad.u-strasbg.fr

Databases in Astronomy




CasJobs Guide - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://casjobs.sdss.org/CasJobs/Guide.aspx
Google
openSUSE Getting Started Latest Headlines
SDSS SkyServer DR6
SkyServer DR6 Search
CasJobs Guide
SDSS Query / CasJobs
Help Tools Create Account Login
Not Logged in
Help
FAQ
Recent Updates
SQL Tutorial
Optimizing Queries
Advanced CasJobs Queries
Developers
Introduction
Welcome to the Catalog Archive Server Jobs System, or CasJobs. This guide assumes you have experience with basic SQL syntax. If you'd like to learn more about SQL there are plenty of other tutorials on the web. This guide will cover the slightly simpler varient of SQL that CasJobs uses.
Registration
Before you can query this system you'll need to register and then log in. Your queries and their resulting data are associated with your account, so don't forget your login! But if lightning strikes, skies fall and you've consequently forgot your password it can be emailed to you from here. Your email may be used to contact you about your account, as well as optionally notifing about query completion. Your UserID will be used to identify you to the group world; should you choose to participate in this please pick a 'nice' UserID. A personal database, or MyDB is also created and assigned to your account on registration.
MyDB
Your MyDB is a personal database created just for you upon registration. You have full privileges in this context; you can create/alter/drop tables, functions and stored procedures. It is designed to be a sort of staging area where you refine your requested data before eventually extracting it to a local copy in CSV, FITS, etc. format. Please note that you can only extract data from MyDB, so any data you will eventually want a local copy of must first be put into MyDB.
Getting Data Into MyDB
Getting data into MyDB is easy. Just write your query, then click submit and it will automatically create a table using the name from the 'Table' field (directly above the query box) in MyDB containing the results of your query. Alternatively, you use an 'into' statement in your query; an example of this is
Done
1 2 4
CasJobs Guide - Mozilla
de
08:09

Database – an example

Basic data :

HD 10123 -- Star

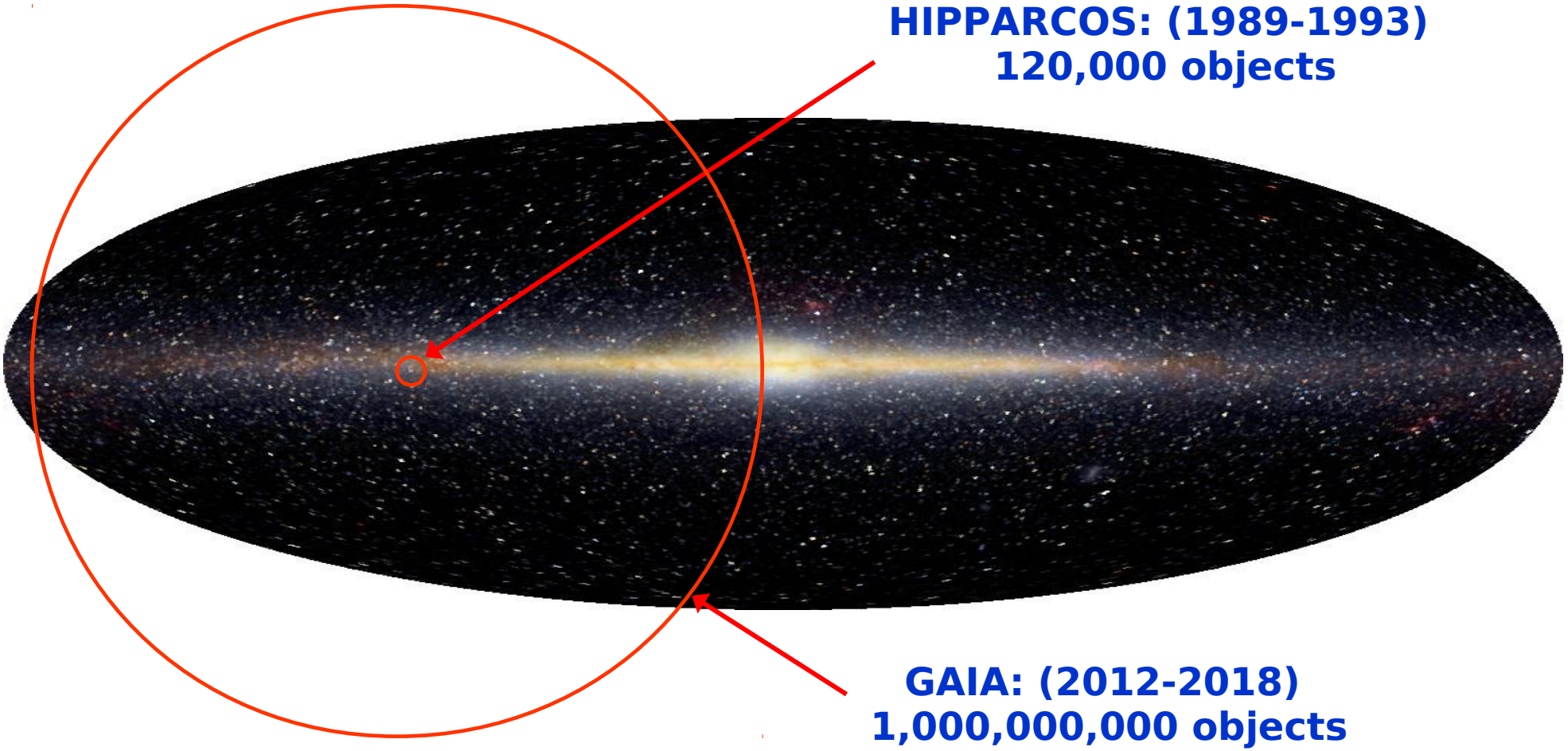
Other object types:	* (HD,CD,CPC,CPD,GEN#,GSC,PPM,TYC) , IR (2MASS)
ICRS coord. (ep=2000) :	023.9888796 -72.8318519 (~) [36.80 23.83 0] B 1998A&A...335L..65H
Proper motions *mas/yr* [error ellipse]:	19.30 32.70 [4.00 2.50 0] B 1998A&A...335L..65H
Spectral type:	K1III C ~
Fluxes (5) :	B 10.01 [~] C ~ V 8.89 [~] C ~ J 6.946 [0.027] C 2003yCat.2246....0C H 6.394 [0.038] C 2003yCat.2246....0C K 6.276 [0.017] C 2003yCat.2246....0C

Identifiers (9) :

HD 10123	CPD-73 93	2MASS J01355732-7249548
CD-73 66	GEN# +1.00010123	PPM 367338
CPC 21.2 223	GSC 09142-00403	TYC 9142-403-1

Astrometry : Distances, Positions & Motions

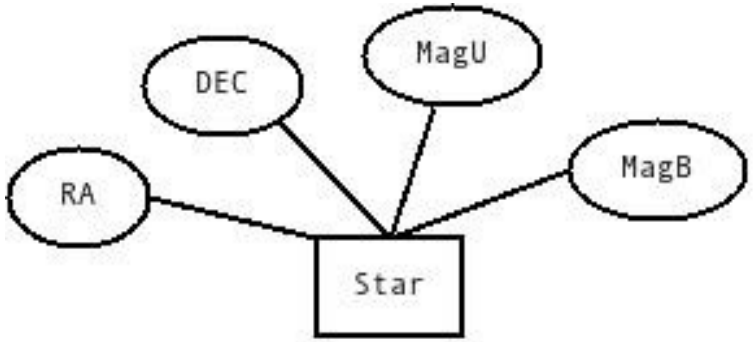
Galaxy : (10^5-10^{10} years)
100,000,000,000 objects

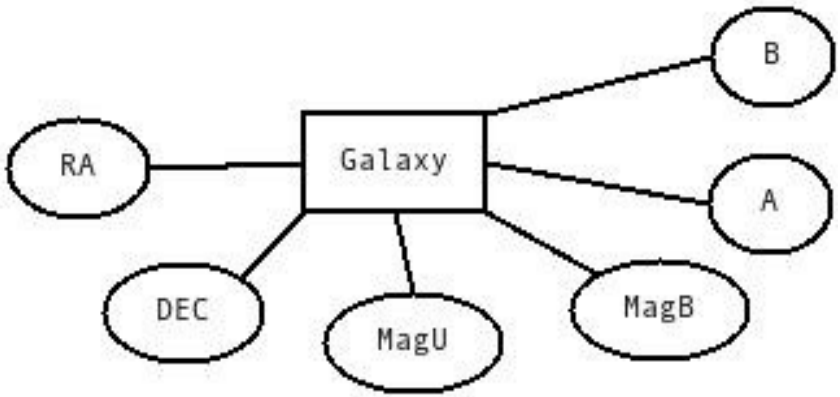


Astronomical catalogs

- Hipparcos: 120,000
- Tycho-2: 1,058,000
- 2MASS-PSC: 471 mln
- USNO-A2: 526 mln
- USNO-B1: 1,045 mln
- SDSS DR12: 1.2 bln

Data Model





- Entities
- Relashionship
- Attributes
- Data types
- Data formats

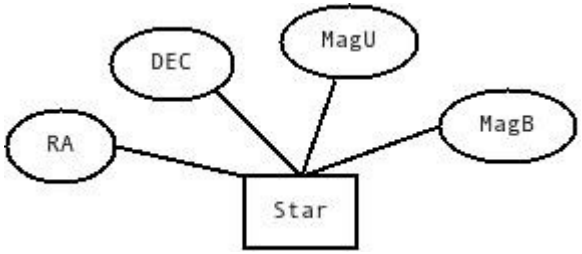
  → Data volume

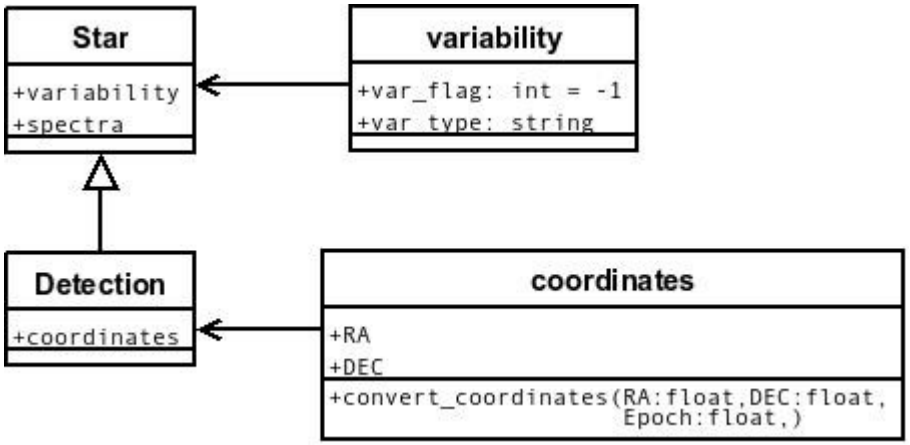
DBMS

- Database Management System
- Implementation of data model
- Storage
- Interfaces

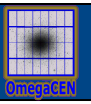
Object-Oriented DBMS

- Object-oriented data model



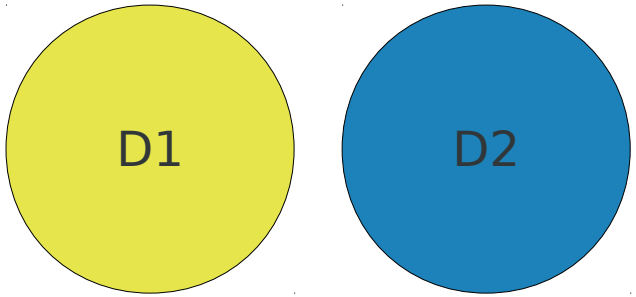


Relational DBMS



- Relational data model
- Relational algebra
- Interfaces of administrator and user
- API in different languages

Relational Algebra



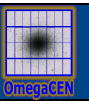
A1=(D1)

{A1,A2,A1,A3,A4}

- domains
- corteges, tuples
- join, intersect, difference
- projection
- products

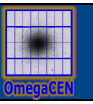
David Maier
The Theory of Relational Data Bases
Financial Times Prentice Hall
1983

Relational Algebra



- Domains (allowed range of values)
- Set
- Attribute
- Relation
- Tuple
- Operation

Relational Algebra Operations



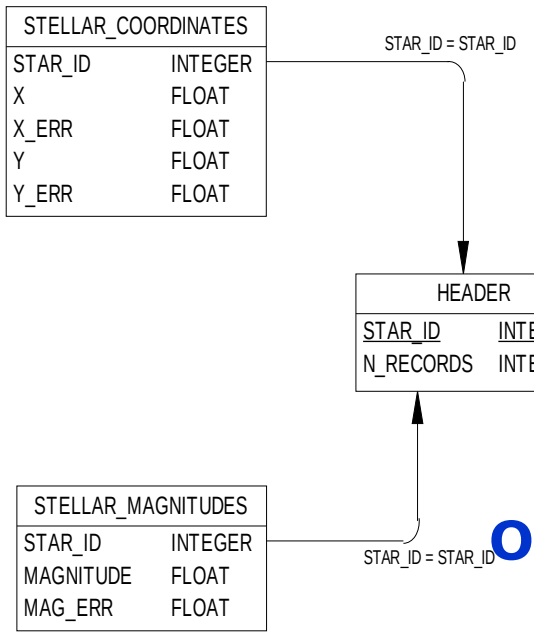
σ	Selection: select all tuples from relation R with some condition
π	Projection: select all tuples with R1 – subset of relation R
ρ	Renaming: change a name for all tuples from R1 to R2
R1xR2	Cross-product : all combinations of rows from R1 and R2
$\bowtie_p$	Theta-Join: make all combination of R1 and R2 and apply condition p
$\bowtie$	Natural Join: make all combination of R1 and R2 and eliminate doubles
U	Union: add R1 and R2
-	Difference:find the same tuples in R1 and R2 and remove them
∩	Intersection: select only tuples which are either in R1 or R2

- relational: Codd, 1972, IBM
- object-oriented: C++, 90s

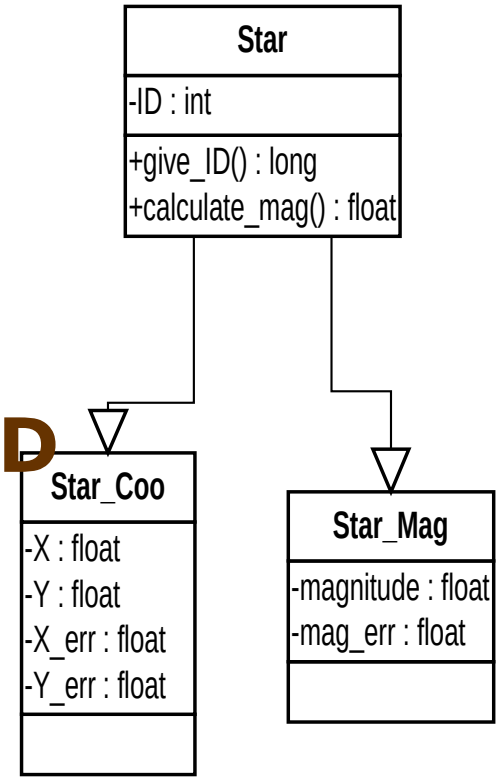
	relational	**object-oriented**
data model	tables, rows, triggers, SPs	classes,objects,methods
applications	4GL, ESQL languages	C++, Java etc
support of objects	limited (SQL3)	native
support of normalization	native	can be implemented
data access	database driven	object driven
data cast	required	not required

RDBMS vs ODBMS

RDBMS



ODBMS



DATA STORAGE

TABLES | **OBJECTS**

REQUEST

by value | **by object ID**

OPTIMIZATION OF SELECTION

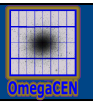
index | **pointer**

Massive storage of simple and homogeneously structured data

Relational DBMS

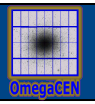
Storage of data with complicated structure

SQL



- language to operate with relations
- standards: SQL89, SQL92, SQL3
- SELECT, INSERT, DELETE, UPDATE
- CREATE, DROP
- DATABASE, TABLE, VIEW, INDEX

SELECT



- form a row/rows with desired restriction
- perform grouping of entities

```
SELECT A, B, C FROM T WHERE A=''

SELECT T1.A, T2.B, T3.C FROM T1, T2 WHERE T1.A=''
        AND T2.A=T1.A ORDER BY T1.A

SELECT NEW_A FROM (SELECT T1.A NEW_A, T2.B, T3.C
         FROM T1, T2 WHERE T1.A=''
        AND T2.A=T1.A)
```

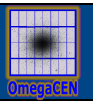
Aggregate functions

- single-value functions: SIN, COS, ...
- aggregate functions: AVG, STDDEV, ...
- grouping of entities essential
- NULL values problem

ID	MAG	MAG_FLAG
101	23.4	0
101	23.3	0
101	21.8	1
102	15.6	2

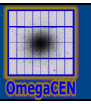
```
SELECT MAX(MAG_DISP) FROM (
    SELECT STDDEV(MAG) MAG_DISP, ID, MAG_FLAG
        FROM T WHERE MAG_FLAG IN [0,1,2]
            GROUP BY ID, MAG_FLAG)
```

Stored Procedures & Triggers



- stored procedures & triggers: user-defined, database level
- stored procedures: functions
- triggers: events (INSERT, SELECT, DELETE)

Indices and Keys



- PK – primary key
- FK – foreign key
- unique index
- indices: ASC and DESC
- check conditions

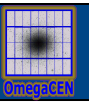
N vs log N

- optimizer
- storage

Data types

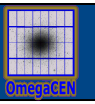
- check your DB!
- numerical: float, double
- DATE
- character: CHAR, CHAR(n), VARCHAR(n)
- BLOBs
- NULL
- UDF

MySQL data types




```
 BIT[(length)]
| TINYINT[(length)] [UNSIGNED]
| SMALLINT[(length)] [UNSIGNED]
| MEDIUMINT[(length)] [UNSIGNED]
| INT[(length)] [UNSIGNED]
| INTEGER[(length)] [UNSIGNED]
| BIGINT[(length)] [UNSIGNED]
| REAL[(length,decimals)] [UNSIGNED]
| DOUBLE[(length,decimals)] [UNSIGNED]
| FLOAT[(length,decimals)] [UNSIGNED]
| DECIMAL[(length[,decimals])] [UNSIGNED]
| NUMERIC[(length[,decimals])] [UNSIGNED] ]
| DATE
| TIME
| TIMESTAMP
| DATETIME
| YEAR
| CHAR[(length)][CHARACTER SET charset_name] [COLLATE collation_name]
| VARCHAR(length)[CHARACTER SET charset_name] [COLLATE collation_name]
| BINARY[(length)]
| VARBINARY(length)
| TINYBLOB
| BLOB
| MEDIUMBLOB
| LONGBLOB
| TINYTEXT [BINARY][CHARACTER SET charset_name] [COLLATE collation_name]
| TEXT [BINARY][CHARACTER SET charset_name] [COLLATE collation_name]
| MEDIUMTEXT [BINARY][CHARACTER SET charset_name] [COLLATE collation_name]
| LONGTEXT [BINARY][CHARACTER SET charset_name] [COLLATE collation_name]
| ENUM(value1,value2,value3,...) [CHARACTER SET charset_name] [COLLATE collation_name]
| SET(value1,value2,value3,...)[CHARACTER SET charset_name] [COLLATE collation_name]
```

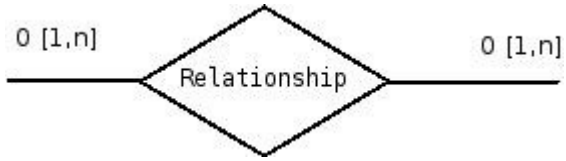
CREATE TABLE



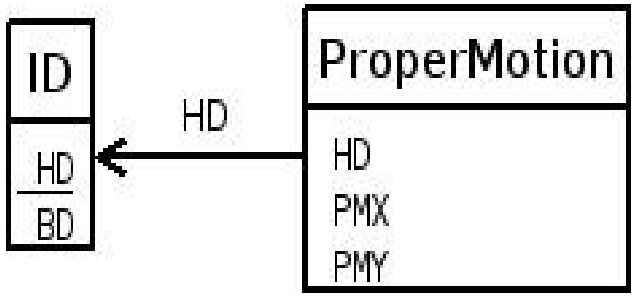

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    { LIKE old_tbl_name | (LIKE old_tbl_name) }
create_definition:
    col_name column_definition
  | [CONSTRAINT [symbol]] PRIMARY KEY [index_type]
(index_col_name,...)
      [index_option] ...
  | {INDEX|KEY} [index_name] [index_type] (index_col_name,...)
      [index_option] ...
  | [CONSTRAINT [symbol]] UNIQUE [INDEX|KEY]
      [index_name] [index_type] (index_col_name,...)
      [index_option] ...
  | {FULLTEXT|SPATIAL} [INDEX|KEY] [index_name]
(index_col_name,...)
      [index_option] ...
  | [CONSTRAINT [symbol]] FOREIGN KEY
      [index_name] (index_col_name,...) reference_definition
  | CHECK (expr)
```

Data Modeling: ER

- Entity-relationship diagram
- Entity: attributes
- Relationship: (smth)-(smth)



Data Modeling: ER




```
CREATE TABLE ID (HD INTEGER NOT NULL,
                 BD INTEGER,
                 CONSTRAINT hd_pk PRIMARY KEY (HD))
CREATE TABLE ProperMotion (HD INTEGER NOT NULL,
                           PMX DOUBLE,
                           PMY DOUBLE,
                           CONSTRAIN hd_fk FOREIN KEY (HD)
                                REFERENCES hd_pk)


SELECT T1.HD, T1.BD, T2.PMX, T2.PMY FROM ID T1, ProperMotion T2 where
                 T2.HD=T1.HD
```

Normalization

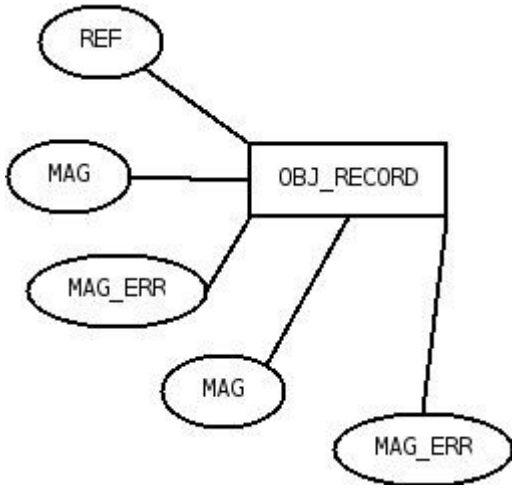
- put the data into the structure
- avoid empty spaces
- 1NF to 5NF

MAG	MAGERR	REF	MAG	MAGERR	REF
15.3	*0.1*	*HD101077,Cross, A&A 123, 23*	*15.2*	*0.05*	*HD101077, Bertinie* AJ 117, 8
14.3	*0.1*	*HD101900,ASCC*	*14.2*	*0.07*	*HD101900, GSC*
18.1	*0.06*	*HD90088,GSC*			

1NF

- the same size for all records
- the table has no double attributes

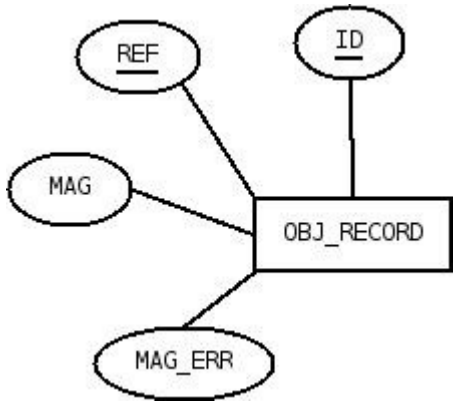
MAG	MAGERR	REF
15.3	*0.1*	*HD101077, Cross, A&A 123, 23*
15.2	*0.05*	*HD101077, Bertinie, AJ 117, 8*
14.3	*0.1*	*HD101900, ASCC*
14.2	*0.07*	*HD101900, GSC*
18.1	*0.06*	*HD90088, GSC*



2NF

- put the key
- the non-key columns describe the key

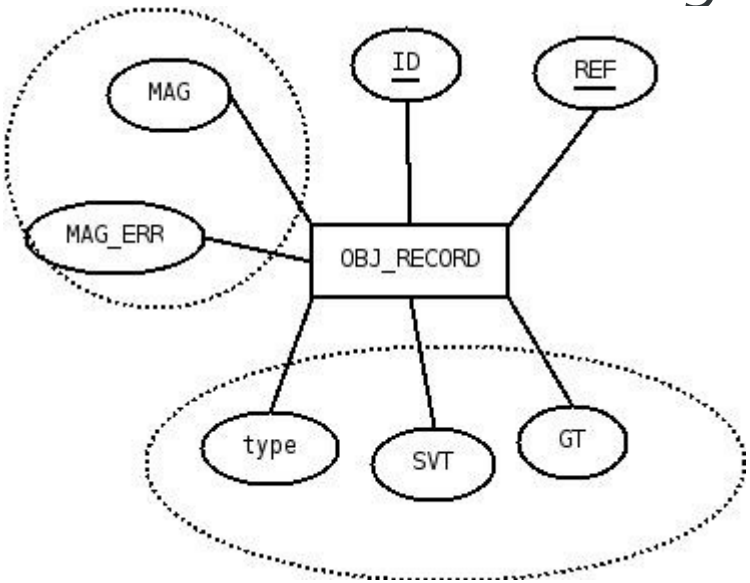
ID	MAG	MAGERR	REF
101077	*15.3*	*0.1*	*Cross, A&A 123, 23*
101077	*15.2*	*0.05*	*Bertinie, AJ 117, 8*
101900	*14.3*	*0.1*	*ASCC*
101900	*14.2*	*0.07*	*GSC*
90088	*18.1*	*0.06*	*GSC*



3NF

- no internal dependences inside the record

ID	REF	MAG	MAGERR	Type	SVT	GT
101077	*1*	*15.3*	*0.1*	*star*	*SB*	
101077	*2*	*15.2*	*0.05*	*star*	*SB*	
101077	*5*			*star*	*SB2*	
101900	*3*	*14.3*	*0.1*	*star*	*CVn*	
101900	*1*	*14.2*	*0.07*	*star*	*CVn*	
90088	*4*	*18.1*	*0.06*	*galaxy*		*SBb*

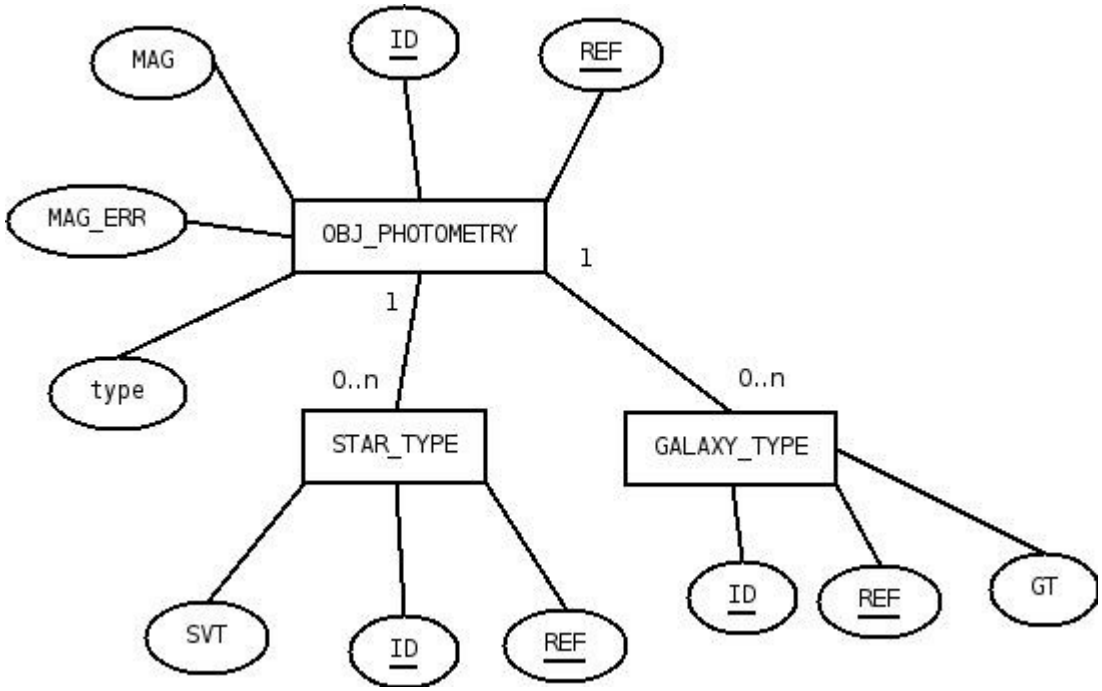


3NF

ID	REF	MAG	MAGERR	Type
101077	1	15.3	0.1	star
101077	2	15.2	0.05	star
101900	3	14.3	0.1	star
101900	1	14.2	0.07	star
90088	4	18.1	0.06	galaxy

ID	REF	SVT
101077	1	SB
101077	2	SB
101077	5	SB2
101900	1	CVn
101900	4	CVn

ID	REF	GT
90088	5	SBb



BCNF

- no transitive dependences

ID	MAG	MAGERR	Type	Filter	LC
101077	*15.3*	*0.1*	*star*	*Johnson V*	*5360*
101077	*15.2*	*0.05*	*star*	*Johnson V*	*5360*
101900	*14.3*	*0.1*	*star*	*SDSS g*	*6000*
101900	*14.2*	*0.07*	*star*	*Johnson V*	*5360*
90088	*18.1*	*0.06*	*galaxy*	*Jonson B*	4950

Filter	LC
SDSS g	*6000*
Johnson V	*5360*
Jonson B	4950

4NF

- removal of 1-n dependencies: only one 1-n dependence per table

ID	SVT	REF	Author
101077	*SB*	*1*	*Cross*
101077	*SB2*	*2*	*Cross*
101900	*CVn*	*3*	*Lejune*
101077	*WA*	*4*	*Lejune*

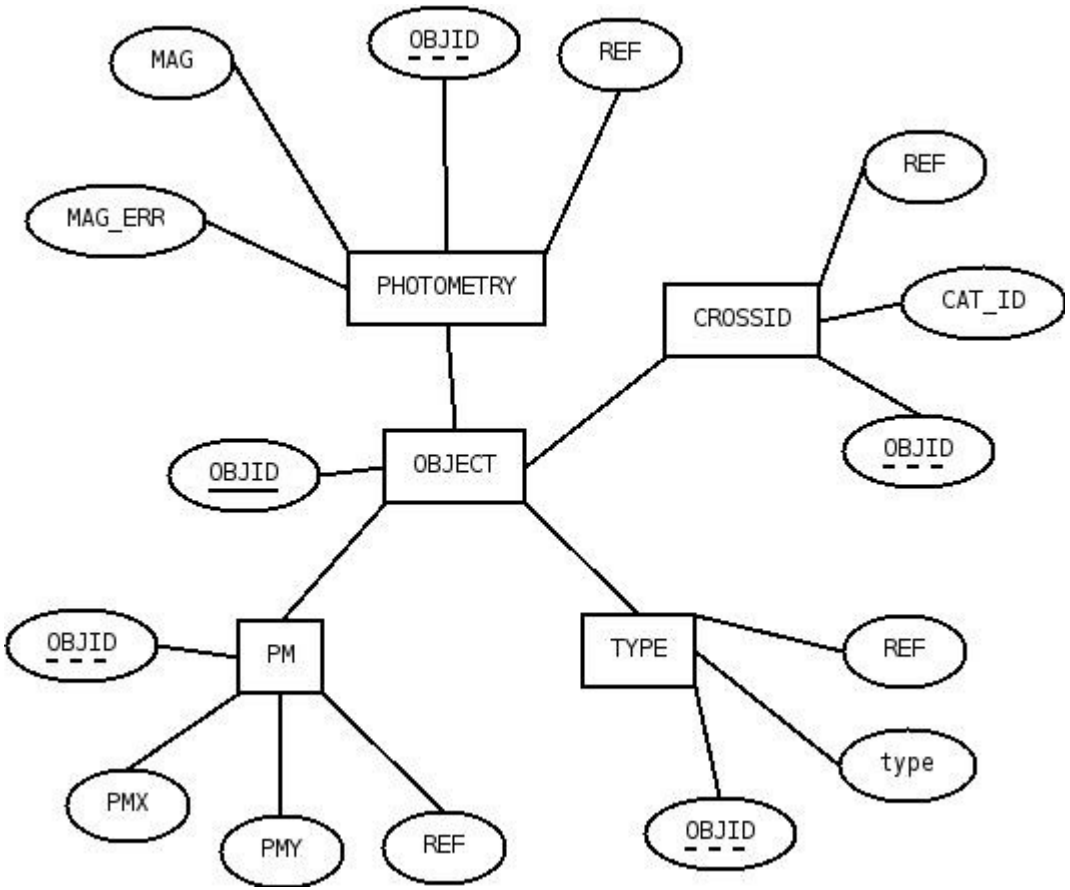
ID	SVT	REF
101077	*SB*	*1*
101077	*SB2*	*2*
101900	*CVn*	*3*
101077	*WA*	*4*

REF	Author
1	*Cross*
2	*Cross*
3	*Lejune*
4	*Lejune*

5NF

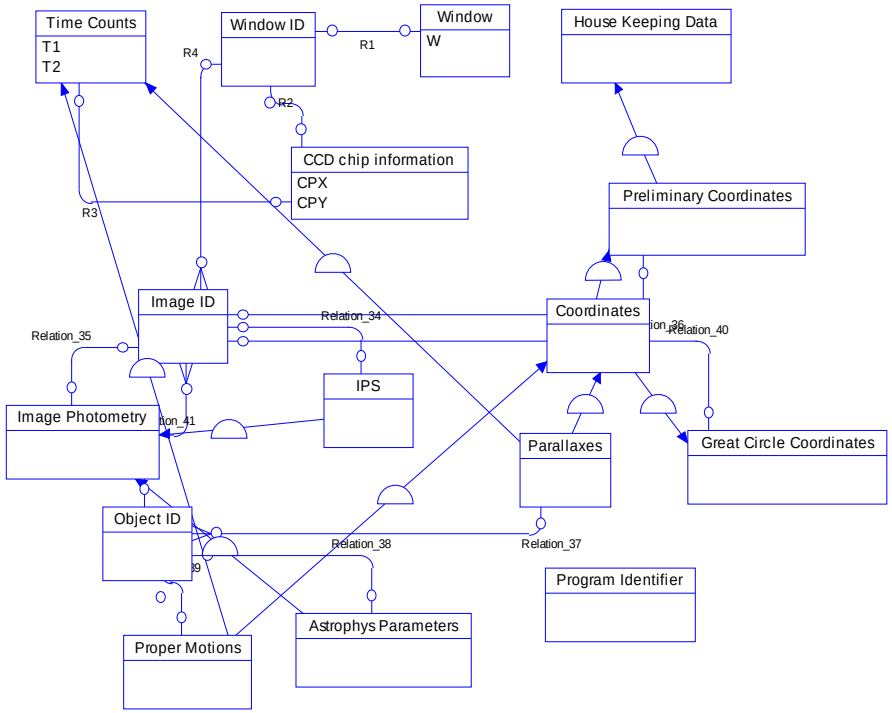
- partitioning of many-to-many relations must be done so, that the recovery of the original table is possible by single SQL request

Star Scheme

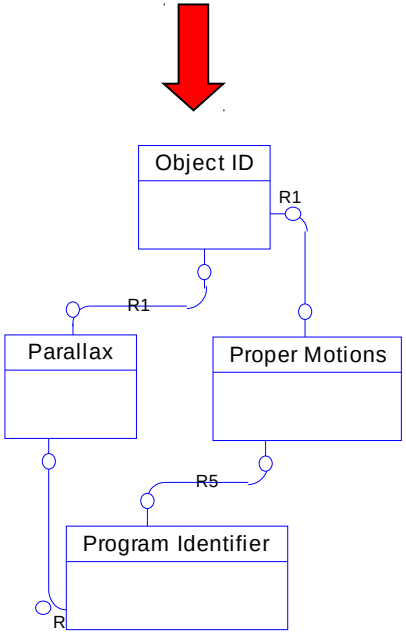


- effective data storage
- no information lost
- ineffective selection for some requests

DB design



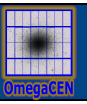
The logical design algorithm



Queries {SELECT; OID, COO, TC;}

- **description for each process**
- **TCF estimation for each operation**
- **final subgraphs**

RDMS: what is available?



- IBM: DB2
- Oracle
- Sybase
- MySQL
- PostgreSQL

Shared-nothing architecture