# Structured Query Language
# SQL Es-Que-El or Sequel

## Didn't we already talk about SQL?

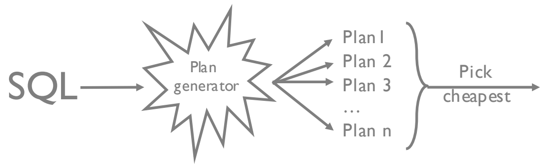
Two sublanguages

**DDL** Data Definition Language
- define and modify schema (physical, logical, view)
- CREATE TABLE, Integrity Constraints

**DML** Data Manipulation Language
- get and modify data
- simple SELECT, INSERT, DELETE
- *human-readable* language

## DBMS (tries to) execute efficiently

Key: precise query semantics
Reorder/modify queries while answers stay same
DBMS estimates costs for different evaluation plans



## SQL: Extended Relational Algebra

Multisets rather than sets
- Relations can contain duplicates (unless constrained)
- Order doesn't matter

NULLs
Aggregates

Most widely used *query language*,
not just relational query language

## Today's Database

Sailors

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 2 | Luis | 2 | 39 |
| 3 | Ken | 8 | 27 |

Boats

| bid | name | color |
|---|---|---|
| 101 | Legacy | red |
| 102 | Melon | blue |
| 103 | Mars | red |

Reserves

| sid | bid | day |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |

Is Reserves table correct?

## Today's Database

Sailors

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 2 | Luis | 2 | 39 |
| 3 | Ken | 8 | 27 |

Boats

| bid | name | color |
|---|---|---|
| 101 | Legacy | red |
| 102 | Melon | blue |
| 103 | Mars | red |

Reserves

| sid | bid | **day** |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |

Is Reserves table correct?
Day should be part of key

## Today's Database

Sailors

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 2 | Luis | 2 | 39 |
| 3 | Ken | 8 | 27 |

Boats

| bid | name | color |
|---|---|---|
| 101 | Legacy | red |
| 102 | Melon | blue |
| 103 | Mars | red |

Reserves

| sid | bid | day |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |
| 2 | 103 | 9/15 |

Is Reserves table correct?
Day should be part of key

## Follow along at home!

```
https://www.instabase.com/ewu/w4111-
public/fs/Instabase%20Drive/Examples/sql.ipynb
```

## <30 year old sailors

```
SELECT *
FROM Sailors
WHERE age < 30
```

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 3 | Ken | 8 | 27 |

```
SELECT name, age
FROM Sailors
WHERE age < 30
```

| name | age |
|---|---|
| Eugene | 22 |
| Ken | 27 |

## <30 year old sailors

```
SELECT *
FROM Sailors
WHERE age < 30
```

$\sigma_{age<30}$ (Sailors)

```
SELECT name, age
FROM Sailors
WHERE age < 30
```

$\pi_{name,\ age}(\sigma_{age<30}$ (Sailors))

## Who reserved boat 102?

Sailors

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 2 | Luis | 2 | 39 |
| 3 | Ken | 8 | 27 |

Reserves

| sid | bid | day |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |

## Who reserved boat 102?

```
SELECT S.name
FROM   Sailors AS S, Reserves AS R
WHERE  S.sid = R.sid AND R.bid = 102
```

Sailors

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 2 | Luis | 2 | 39 |
| 3 | Ken | 8 | 27 |

Reserves

| sid | bid | day |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |

| name |
|---|
| Eugene |
| Luis |

## Who reserved boat 102?

```
SELECT S.name
FROM   Sailors AS S, Reserves AS R
WHERE  S.sid = R.sid AND R.bid = 102
```

(equi-join)

$\pi_{name}(\sigma_{bid=102}(Sailors \bowtie_{sid} Reserves))$

## Who reserved boat 102?

```
SELECT S.name
FROM   Sailors AS S, Reserves AS R
WHERE  S.sid = R.sid AND R.bid = 102
```

Sailors

| sid | name | rating | age |
|---|---|---|---|
| 1 | Eugene | 7 | 22 |
| 2 | Luis | 2 | 39 |
| 3 | Ken | 8 | 27 |

Reserves

| sid | bid | day |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |
| **1** | **102** | **9/15** |

| name |
|---|
| Eugene |
| Luis |
| Eugene |

## DISTINCT: unique rows / set

Reserves

| sid | bid | day |
|---|---|---|
| 1 | 102 | 9/12 |
| 2 | 102 | 9/13 |
| 2 | 103 | 9/14 |

```
SELECT  bid
FROM    Reserves
```

| bid |
|---|
| 102 |
| 102 |
| 103 |

```
SELECT  DISTINCT bid
FROM    Reserves
```

| bid |
|---|
| 102 |
| 103 |

## Structure of a SQL Query

DISTINCT
Optional: Remove duplicates (set)
Default: duplicates permitted (multiset)

target-list
List of expressions over attrs of tables in relation-list
e.g., SELECT s.name

SELECT [DISTINCT] *target-list*
FROM *relation-list*
WHERE *qualification*

relation-list
List of relation names
Can define aliases "AS X"
e.g., FROM sailors AS s, reserves as R

qualification
Boolean expressions
Combined w/ AND,OR,NOT
attr *op* const
$attr_1$ *op* $attr_2$
op is =,<,>,<>,etc

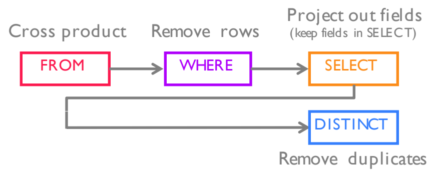
## Semantics

SELECT [DISTINCT] *target-list*
FROM *relation-list*
WHERE *qualification*

| | |
|---|---|
| FROM | compute cross product of relations |
| WHERE | remove tuples that fail qualifications |
| SELECT | remove fields not in target-list |
| DISTINCT | remove duplicate rows |

## Conceptual Query Evaluation

SELECT [DISTINCT] *target-list*
FROM *relation-list*
WHERE *qualification*
*GROUP BY grouping-list*
*HAVING group-qualification*



Not how actually executed! Above is likely very slow

## Sailors that reserved 1+ boats

```
~~SELECT  S.sid~~
~~FROM    Sailors AS S, Reserves AS R~~
~~WHERE   S.sid = R.sid~~
```

Would DISTINCT change anything in this query?

Sailors.sid is a primary key

What if SELECT clause was SELECT S.name?

## Sailors that reserved 1+ boats

```
SELECT  DISTINCT S.sid
FROM    Sailors AS S, Reserves AS R
WHERE   S.sid = R.sid
```

## Table Alias (AS, Range Variables)

Disambiguate relations

same table used multiple times (self join)

```
~~SELECT  sid~~
~~FROM    Sailors, Sailors~~
~~WHERE   age > age~~

SELECT  S1.sid
FROM    Sailors AS S1, Sailors AS S2
WHERE   S1.age > S2.age
```

## Table Alias (AS, Range Variables)

Disambiguate relations

same table used multiple times (self join)

```
~~SELECT  sid~~
~~FROM    Sailors, Sailors~~
~~WHERE   age > age~~

SELECT  S1.name, S1.age, S2.name, S2.age
FROM    Sailors AS S1, Sailors AS S2
WHERE   S1.age > S2.age
```

## Expressions (Math)

```
SELECT  S.age, S.age – 5 AS age2, 2*S.age AS age3
FROM    Sailors AS S
WHERE   S.name = ‘eugene’

SELECT  S1.name AS name1, S2.name AS name2
FROM    Sailors AS S1, Sailors AS S2
WHERE   S1.rating*2 = S2.rating - 1
```

## Expressions (Strings)

```
SELECT  S.name
FROM    Sailors AS S
WHERE   S.name LIKE ‘e_%’
```

Strings quoted with single quotes:‘ (identifiers: double quote)

If you need an embedded quote: use two: 'this is "quoted" '

| | |
|---|---|
| ‘_’ | any one character (• in regex) |
| ‘%’ | 0 or more characters of any kind (•* in regex) |

Most DBMSes have rich string manipulation support e.g., regex

PostgreSQL documentation

http://www.postgresql.org/docs/9.3/static/functions-string.html

## Expressions (Date/Time)

```
SELECT  R.sid
FROM    Reserves AS R
WHERE   now() - R.date < interval '1 day'
```

TIMESTAMP, DATE, TIME types
Values quoted: '2016-02-16', 'Feb-16-2016', '4:05 PM'
now() returns timestamp at start of transaction
DBMSes provide rich time manipulation support
exact support may vary by vender

Postgresql Documentation
http://www.postgresql.org/docs/9.3/static/functions-datetime.html

## Expressions

| | |
|---|---|
| Constant | 1, 'hello', 7.85 |
| Col reference | Sailors.name |
| Arithmetic | Sailors.sid * 10 |
| Unary operators | NOT |
| Binary operators | AND, OR, <, =, <>, >= |
| Function | abs(), sqrt(), ... |
| Casting | 1.7::int, '10-12-2015'::date |

## UNION, INTERSECT, EXCEPT

Algebra: ∪, ∩, -

Combine results from two queries:

SELECT [query1] UNION SELECT [query2]

By default: *distinct results!* (set semantics)
*[operator]* ALL: Keep duplicates: multi-set

SELECT [query 1] UNION ALL SELECT [query2]