



*Don Chamberlain, en av skaparna av SQL*

# SQL

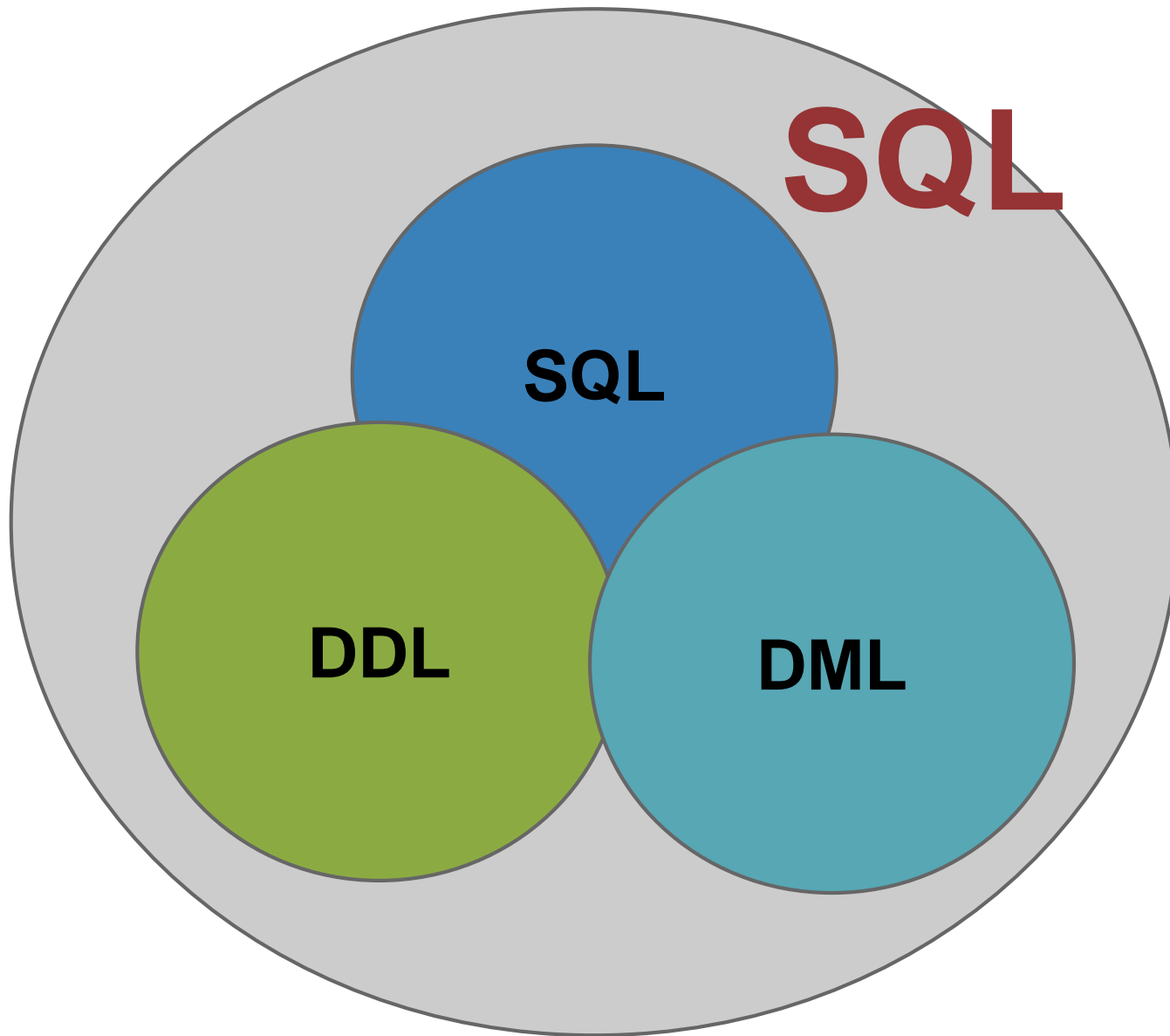
Structured Query Language

# Agenda

- SQL
  - vad är det
  - hur går det till
  - SELECT
    - grundläggande syntax
    - equi joins
  - functions
  - INSERT
  - UPDATE
  - DELETE

# SQL

- Ett språk för att kommunicera med en RDB.
- SQL är egentligen ett av flera språk som vi använder mot en RDB.



# SQL

- Ett samlingsnamn för alla språk är SQL.  
Men, som sagt, språken som ingår är;
- **SQL**  
hämta data
- **DDL**  
skapa data, tabeller, databaser
- **DML**  
manipulera data, databaser, tabeller

# SQL

- **Statements/kommandon**

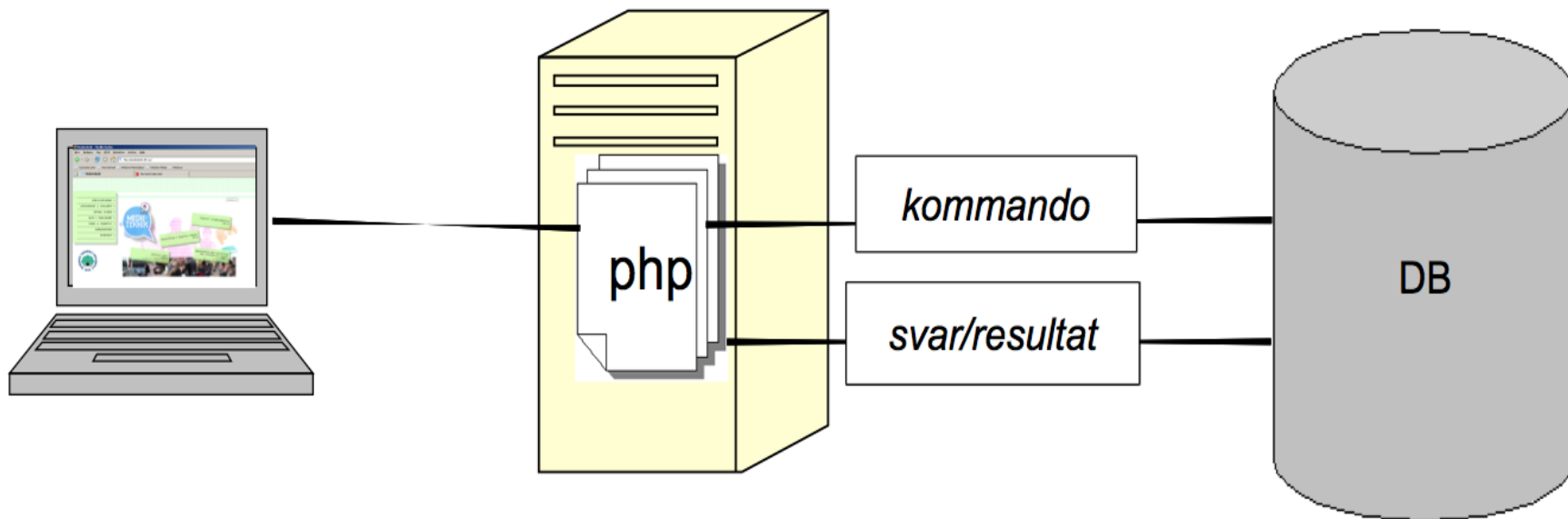
För att kommunicera och utföra operationer med och mot en databas konstruerar man s.k SQL-statements/kommandon.

- Beroende på kommandots innehåll så kan vi få ett svar från databasen. Detta svar kan vi ta hand om och utföra operationer i applikationen med.

# SQL

- Ett kommando konstrueras i;
  - en MySql konsoll (vi arbetar direkt i RDB)
  - från en applikation (skriven i ett programspråk)
  - ett RDBMS, t ex phpMyAdmin





# SQL

- För att hämta data och presentera denna används kommandot SELECT.
- Beroende på vad vi vill hämta kan vi formulera korta enkla kommandon till långa och väldigt komplexa.

- **SYNTAX:**

```
SELECT [DISTINCT] columnA, columnB, ...  
                                FROM table1, table2, ...  
[WHERE condition expr]  
[GROUP BY column list]  
[HAVING conditional expr]  
[ORDER BY column list]
```

**Tex:** SELECT Name FROM Person;

# SELECT

- `SELECT kolumn1 [, ..., ...] FROM tabell1, t2`
- När man exekverar ett kommando av typen `SELECT` hämtar man data från tabeller. Resultatet man får tillbaka kallas *resultat*, *recordset* eller *svarstabell*.
- Resultatet tar vi hand om, tolkar och eventuellt presenterar på t ex en webbsida (med php etc).

# Exempel databas

Person

<b><u>PersonId</u></b>	<b>Firstname</b>	<b>Lastname</b>
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson

# SELECT k1, k2, k3, kn, ... FROM Tabell

```
SELECT Firstname, Lastname FROM  
Person;
```



<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson



Firstname	Lastname
Lars	Larsson
Lisa	Svensson
Kajsa	Karlsson

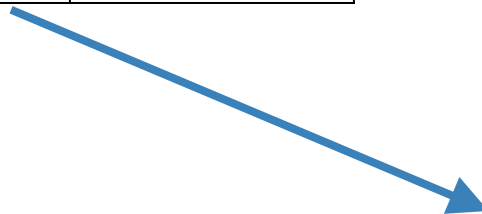
Svarstabell som vi får tillbaka.

Om kommandot är exekverat via php får vi tillbaka denna tabell (jmf med en array) som vi kan arbeta med.

# SELECT med WHERE klausul

```
SELECT Firstname, Lastname FROM  
Person  
WHERE Firstname="Lars";
```

<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson



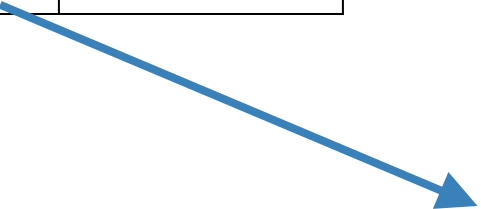
Firstname	Lastname
Lars	Larsson

# SELECT med WHERE klausul 2

```
SELECT Firstname, Lastname FROM  
Person  
WHERE PersonId=1;
```



<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson



Firstname	Lastname
Lars	Larsson

# SELECT med WHERE och OR

```
SELECT Firstname, Lastname FROM  
Person  
WHERE Firstname="Lars"  
OR Lastname="Svensson";
```



<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson



Firstname	Lastname
Lars	Larsson
Lisa	Svensson



# SELECT med WHERE och AND 2

```
SELECT Firstname, Lastname FROM Person  
WHERE Firstname="Lars"  
AND Lastname="Svensson" AND PersonId=1;
```



<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson



# SELECT med WHERE, AND och OR

```
SELECT Firstname, Lastname FROM  
Person  
WHERE Firstname="Lars"  
OR Lastname="Svensson";
```

Svarstabell - se föregående slide.

AND och OR kan kombineras i oändliga kombinationer.

Jmfr AND med logisk &&  
Jmfr OR med logisk ||

# Operatorer

- Det finns ett antal operatorer för jämförelser <, >, <=, >=

*Ex: SELECT FirstName FROM Employee WHERE Salary >= 10000*

- BETWEEN

*Ex: SELECT Name FROM Employee WHERE Salary BETWEEN 20000 AND 13000*

- IN, NOT IN

*Ex: SELECT \* FROM Person WHERE FirstName IN ('Svensson', 'Andersson')*

- IS NULL, IS NOT NULL

*Ex: SELECT \* FROM Books WHERE StudentID IS NOT NULL*

# SELECT... WHERE... LIKE

```
SELECT LastName, FirstName FROM Person  
WHERE Lastname LIKE "s%"
```



Firstname	Lastname
Lars	Larsson
Lisa	Svensson

# LIKE med wildcards - sökning

- `...WHERE FirstName LIKE "%a"`

Alla förnamn som slutar på tecknet a.

- `...WHERE LastName LIKE "s_ensson`

1:a tecket ska vara 's', 2:a tecknet obestämt, sluta med ensson.

- `...WHERE LastName LIKE "s_e%"`

Första tecknet 's', andra obestämt, tredje e, resten obestämda och hur många som helst.

# forts LIKE

- LIKE är inte case sensitive.

- LIKE fungerar även med siffror

```
SELECT nbr FROM t1 WHERE nbr LIKE '1%';
```

- NOT framför LIKE negerar uttrycket

```
SELECT LastName, FirstName FROM Person  
WHERE Lastname NOT LIKE "s%";
```

- Escape tecken är \

```
SELECT path FROM t1 WHERE path LIKE '\\\%';
```

- Det finns mer om LIKE... läs dokumentationen.

# FUNCTIONS

- I RDB's finns färdiga funktioner att tillgå och använda i ett SQL-statement.
- Funktionerna kan summera tal, addera, ta reda på aktuellt klockslag.... ja lite allt möjligt.
- Funktioner är specifika hos respektive RDB.
- I MySql (oftast);  
NAMN\_PÅ\_FUNKTION ( [p1, p2, pN] )

- I MySql finns många;  
<http://dev.mysql.com/doc/refman/5.6/en/func-op-summary-ref.html>

# FUNCTIONS - några exempel

```
SELECT LOWER("MAJS"); --- majs
```

```
SELECT NOW(); --- YYYY-MM-DD HH:MM:SS
```

```
SELECT SUM(nbr) FROM t1;  
--- summan av alla nbrs värden
```

```
SELECT SHA2('password', 12);  
--- krypterad enl SHA2 med längden 12
```

```
SELECT SUBSTRING('foobar', 4); --- bar
```

```
M FL, M FL
```



# Slutligen om FUNCTIONS

Ta en titt i dokumentationen - där finns alltid en funktion för någonting...

**JULIAN GRAVES LTD** ©

KINGSWINFORD, WEST MIDLANDS, DY6 7FT

## MILK CHOCOLATE COATED RAISINS

Milk Chocolate Contains Vegetable Fat In Addition To Cocoa Butter  
Cocoa Solids 20% Minimum, Milk Solids 20% Minimum

Ingredients: Milk Chocolate (54%) (Sugar, Skimmed Milk Powder, Cocoa  
Butter, Cocoa Mass, Butter Oil, Lactose, Vegetable Oil, Whey Powder,  
Emulsifier: Soya Lecithin; Flavouring); Raisins (45%).

SELECT \* FROM [Equipment Table] WHERE [Equipment ID]=4;

Packed In An Environment Where Gluten, Nuts & Sesame Seeds May Be Present

**200ge**

**BEST BEFORE END**

**AUG 2007**

**6317T4A**



# EQUI JOINS

**Meh!!1**

Om man vill hämta data från 2st tabeller då?

# Exempel databas – IMPLEMENTERA DEN

Person

<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson

Address

<u>AddressId</u>	PersonId	Zip	Street	City
1	1	12345	Broadway 1	New York
2	2	12345	Broadway 45	New York
3	1	12332	Gatan 2	Kalmar
4	2	12345	Broadway 1	New York
5	3	12345	Broadway 1	New York

# EQUI JOINS

- Vilka personer bor på vilka adresser?
- Datat finns i 2st tabeller.... gah!!!1
- EQUI JOINS to the rescue.
- Vi använder oss av PK och FK i en WHERE-klausul. Vi 'länkar' ihop dem.

# Exempel databas

Person

<u>PersonId</u>	Firstname	Lastname
1	Lars	Larsson
2	Lisa	Svensson
3	Kajsa	Karlsson

Address

<u>AddressId</u>	PersonId	Zip	Street	City
1	1	12345	Broadway 1	New York
2	2	12345	Broadway 45	New York
3	1	12332	Gatan 2	Kalmar
4	2	12345	Broadway 1	New York
5	3	12345	Broadway 1	New York

# EQUI JOINS

```
SELECT t1.k1, t2.k2 FROM t1, t2  
WHERE t1.PK = t2.FK;
```

# EQUI JOINS

```
SELECT Person.Firstname, Address.Street,  
Address.City FROM Person, Address  
WHERE Person.PersonId = Address.PersonId;
```



Firstname	Street	City
Lars	Broadway 1	New York
Lars	Gatan 2	Kalmar
Lisa	Broadway 45	New York
Lisa	Broadway 1	New York
Kajsa	Broadway 1	New York



# EQUI JOINS

Svårare än så är det inte.

LÄNKA IHOP FLER TABELLER I WHERE-  
KLAUSULEN MED PK=FK

# ALIAS för tabeller

```
SELECT P.Firstname, A.Street, A.City  
FROM Person P, Address A  
WHERE P.PersonId = A.PersonId;
```



Firstname	Street	City
Lars	Broadway 1	New York
Lars	Gatan 2	Kalmar
Lisa	Broadway 45	New York
Lisa	Broadway 1	New York
Kajsa	Broadway 1	New York

# ALIAS för kolumner

```
SELECT P.Firstname AS Namn,  
A.Street AS Gata, A.City AS Stad  
FROM Person P, Address A  
WHERE P.PersonId = A.PersonId;
```



Namn	Gata	Stad
Lars	Broadway 1	New York
Lars	Gatan 2	Kalmar
Lisa	Broadway 45	New York
Lisa	Broadway 1	New York
Kajsa	Broadway 1	New York

# **EQUI JOINS going crazy**

Om vi har 3st tabeller?

En sån däringa mellantabell?

Hur hämtar vi ut data ur 3 tabeller? Crazy.

No problem my friend.

Länka nycklarna....

<b><u>PersonId</u></b>	<b>Surname</b>	<b>Lastname</b>	Person
1	Lars	Larsson	
2	Lisa	Svensson	
3	Kajsa	Karlsson	

<b>PersonId</b>	<b>AddressID</b>	PersonAddress
1	1	
2	1	
3	3	
1	2	

<b><u>AddressId</u></b>	<b>Zip</b>	<b>Street</b>	<b>City</b>	Address
1	12345	Broadway 1	New York	
2	12345	Broadway 45	New York	
3	12332	Gatan 2	Kalmar	

```
SELECT P.Firstname, A.Street  
FROM Person P, Address A, PersonAddress PA  
WHERE P.PersonId=PA.PersonId  
AND PA.AddressId = A.AddressId;
```

**--- Bara Lars adresser:**

```
SELECT P.Firstname, A.Street FROM Person P,  
Address A, PersonAddress PA  
WHERE P.PersonId=PA.PersonId  
AND PA.AddressId = A.AddressId  
AND P.PersonId=1;
```

# INSERT - skapa data

```
INSERT INTO t VALUES (x, 'y', z);  
-- strängar inom ' eller ''
```

```
INSERT INTO Person(12, 'Kent',  
'Svensson');
```

```
---Alt:
```

```
INSERT INTO Person (Firstname)  
VALUES ('Leif');
```

# DELETE - ta bort data

```
DELETE FROM t [WHERE...]
```

```
DELETE FROM Person;
```

```
--- tömmer Person på all data
```

```
DELETE FROM Person WHERE PersonId=1;
```

```
--- tar bort Lars...
```



# UPDATE - uppdatera befintlig data

```
UPDATE t SET col1=val[, colN=val [WHERE...]]
```

```
UPDATE Person SET Firstname='Leif';  
--- samtliga personers förnamn blir Leif
```

```
UPDATE Person SET Firstname='Leif',  
Lastname='Olofsson' WHERE PersonId=1;  
--- Lars byter för- och efternamn
```

tack