



# SQL Queries

# Key

A key is a unique column that cannot be null

The database creates an **index** for keys which make it much faster to retrieve a row based on its key.

Example:

```
select *  
from student  
where email= "cph-rg54@cphbusiness.dk"
```

Sometimes a key is composite

Example:

```
CREATE TABLE orderdetails (  
  orderNumber int(11) NOT NULL,  
  productCode varchar(15) NOT NULL,  
  quantityOrdered int(11) NOT NULL,  
  priceEach decimal(10,2) NOT NULL,  
  orderLineNumber smallint(6) NOT NULL,  
  PRIMARY KEY (orderNumber,productCode)  
)
```

# Primary Key

A primary key is a key chosen for identification of rows in a table

You should not change a primary key over time.

This means that phone numbers and email addresses are poor primary keys

# Foreign Key

A foreign key is a column that has a value which exists as primary keys in a (other) table

Example: The **office** table in the classicmodels database has primary key "officeCode". In the table **employees**, there is a column which represents the office that the employee belongs to. This column is a foreign key.

Notice: Since more employees can belong to the same office, the foreign key is not unique in the **employees** table. Also, freelancers do not belong to an office and therefore the foreign key might also be null.

# Enforce referential integrity between the tables

- A “**foreign key**” links each occurrence in a relation representing a child entity to the occurrence of the parent entity containing the matching candidate (usually primary) key
- Referential Integrity means that if the foreign key contains a value, that value must refer to an existing occurrence in the parent entity
- What is the constraints between **offices** and **employees** tables
  - Parent child relationship

# Revision of Day 1

- Forward Engineering
- Reverse Engineering

# Tuesday – SQL Queries

- We will recap the SQL select statement to query a database.
- We will focus on multi-table queries (joins) and SQL aggregate functions (especially grouping of results)

# Resources

There are different resources about SQL and MySQL online. Three good ones are:

- <http://www.w3schools.com/sql/default.asp>
- <http://www.mysqltutorial.org/basic-mysql-tutorial.aspx>

Wikipedia also has good examples for most SQL commands



# Database Table Structure

## Columns

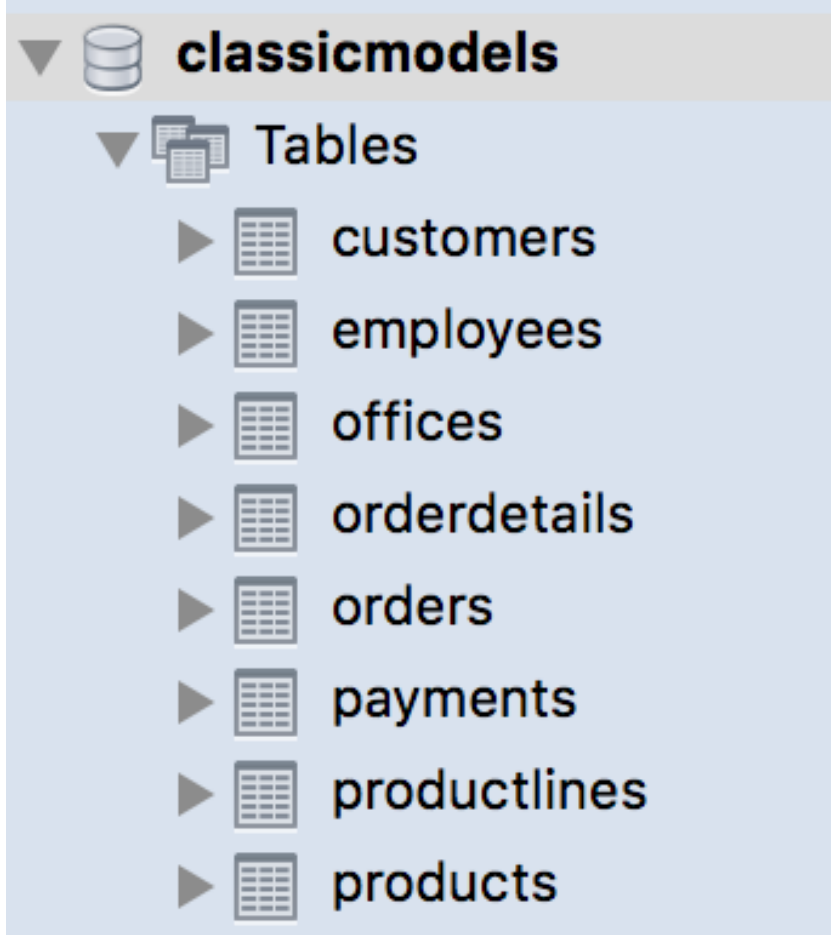
## Rows

productCo...	productName	productLine	productScale	productVendor	productDescription	quantityInSto...	buyPrice	MSRP
S10_4757	1972 Alfa Romeo GTA	Classic Cars	1:10	Motor City Art Classics	Features include: Turnable front wheels; steerin...	3252	85.68	136.00
S10_4962	1962 LanciaA Delta 16V	Classic Cars	1:10	Second Gear Diecast	Features include: Turnable front wheels; steerin...	6791	103.42	147.74
S12_1099	1968 Ford Mustang	Classic Cars	1:12	Autoart Studio Design	Hood, doors and trunk all open to reveal highly...	68	95.34	194.57
S12_1108	2001 Ferrari Enzo	Classic Cars	1:12	Second Gear Diecast	Turnable front wheels; steering function; detaile...	3619	95.59	207.80
S12_1666	1958 Setra Bus	Trucks and Buses	1:12	Welly Diecast Productions	Model features 30 windows, skylights & glare re...	1579	77.90	136.67
S12_2823	2002 Suzuki XREO	Motorcycles	1:12	Unimax Art Galleries	Official logos and insignias, saddle bags located...	9997	66.27	150.62
S12_3148	1969 Corvair Monza	Classic Cars	1:18	Welly Diecast Productions	1:18 scale die-cast about 10" long doors open,...	6906	89.14	151.08
S12_3380	1968 Dodge Charger	Classic Cars	1:12	Welly Diecast Productions	1:12 scale model of a 1968 Dodge Charger. Ho...	9123	75.16	117.44
S12_3891	1969 Ford Falcon	Classic Cars	1:12	Second Gear Diecast	Turnable front wheels; steering function; detaile...	1049	83.05	173.02
S12_3990	1970 Plymouth Hemi Cuda	Classic Cars	1:12	Studio M Art Models	Very detailed 1970 Plymouth Cuda model in 1:1...	5663	31.92	79.80
S12_4473	1957 Chevy Pickup	Trucks and Buses	1:12	Exoto Designs	1:12 scale die-cast about 20" long Hood opens,...	6125	55.70	118.50
S12_4675	1969 Dodge Charger	Classic Cars	1:12	Welly Diecast Productions	Detailed model of the 1969 Dodge Charger. Thi...	7323	58.73	115.16
S18_1097	1940 Ford Pickup Truck	Trucks and Buses	1:18	Studio M Art Models	This model features soft rubber tires, working st...	2613	58.33	116.67
S18_1129	1993 Mazda RX-7	Classic Cars	1:18	Highway 66 Mini Classics	This model features, opening hood, opening do...	3975	83.51	141.54
S18_1342	1937 Lincoln Berline	Vintage Cars	1:18	Motor City Art Classics	Features opening engine cover, doors, trunk, an...	8693	60.62	102.74
S18_1367	1936 Mercedes-Benz 500K Special Ro...	Vintage Cars	1:18	Studio M Art Models	This 1:18 scale replica is constructed of heavy d...	8635	24.26	53.91
S18_1589	1965 Aston Martin DB5	Classic Cars	1:18	Classic Metal Creations	Die-cast model of the silver 1965 Aston Martin...	9042	65.96	124.44
S18_1662	1980s Black Hawk Helicopter	Planes	1:18	Red Start Diecast	1:18 scale replica of actual Army's UH-60L BLA...	5330	77.27	157.69
S18_1749	1917 Grand Touring Sedan	Vintage Cars	1:18	Welly Diecast Productions	This 1:18 scale replica of the 1917 Grand Tourin...	2724	86.70	170.00
S18_1889	1948 Porsche 356-A Roadster	Classic Cars	1:18	Gearbox Collectibles	This precision die-cast replica features opening...	8826	53.90	77.00
S18_1984	1995 Honda Civic	Classic Cars	1:18	Min Lin Diecast	This model features, opening hood, opening do...	9772	93.89	142.25
S18_2238	1998 Chrysler Plymouth Prowler	Classic Cars	1:18	Gearbox Collectibles	Turnable front wheels; steering function; detaile...	4724	101.51	163.73
S18_2248	1911 Ford Town Car	Vintage Cars	1:18	Motor City Art Classics	Features opening hood, opening doors, opening...	540	33.30	60.54
S18_2319	1964 Mercedes Tour Bus	Trucks and Buses	1:18	Unimax Art Galleries	Exact replica. 100+ parts. working steering syst...	8258	74.86	122.73
S18_2325	1932 Model A Ford J-Coupe	Vintage Cars	1:18	Autoart Studio Design	This model features grille-mounted chrome horn...	9354	58.48	127.13
S18_2432	1926 Ford Fire Engine	Trucks and Buses	1:18	Carousel DieCast Legends	Gleaming red handsome appearance. Everythin...	2018	24.92	60.77

Columns have a name and a simple type (integer, text, date, boolean, ...)

Rows contain related data – hereby its name “relational database”.

# Database & Tables



A database has a name  
(here “classicmodels”)

A database schema can have  
more than one table

Each table has a unique  
name in the database

# SQL – Standard Query Language

Two type of commands




- Commands that do not change the database
- Commands that change the database

To retrieve data in the database, we use the command (we can use lowercase or uppercase):

```
select ... from ... where ...
```

# Demo

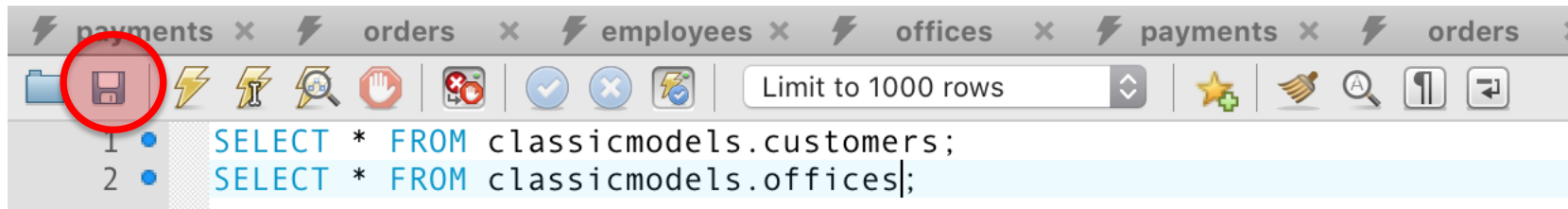
```
SELECT productName, buyPrice, productDescription
FROM products
WHERE buyPrice < 30;
```

Result Grid   Filter Rows: <input type="text" value="Search"/> Export: 			
productName	^	buyPrice	productDescription
1926 Ford Fire Engine		24.92	Gleaming red handsome appearance. Everythin...
1930 Buick Marquette Phaeton		27.06	Features opening trunk, working steering system
1936 Harley Davidson El Knucklehead		24.23	Intricately detailed with chrome accents and trim...
1936 Mercedes Benz 500k Roadster		21.75	This model features grille-mounted chrome horn...
1936 Mercedes-Benz 500K Special Roadster		24.26	This 1:18 scale replica is constructed of heavy d...
1937 Horch 930V Limousine		26.30	Features opening hood, opening doors, opening...
1938 Cadillac V-16 Presidential Limousine		20.61	This 1:24 scale precision die cast replica of the...
1939 Cadillac Limousine		23.14	Features completely detailed interior including V...
1939 Chevrolet Deluxe Coupe		22.57	This 1:24 scale die-cast replica of the 1939 Che...
1950's Chicago Surface Lines Streetcar		26.72	This streetcar is a joy to see. It has 80 separate...
1954 Greyhound Scenicruiser		25.98	Model features bi-level seating, 50 windows, sk...
1958 Chevy Corvette Limited Edition		15.91	The operating parts of this 1958 Chevy Corvette...
1966 Shelby Cobra 427 S/C		29.18	This diecast model of the 1966 Shelby Cobra 42...
1982 Ducati 996 R		24.14	Features rotating wheels , working kick stand. C...
1982 Lamborghini Diablo		16.24	This replica features opening doors, superb det...
Corsair F4U ( Bird Cage)		29.34	Has retractable wheels and comes with a stand....

# Exercises

1. Install database on your own computer by running SQL script "classicmodels.sql"  
File → Run SQL Script → enter file name → enter name of database schema (name it classicmodels)
2. SQL queries:
  - A. Find customers from France
  - B. Find first name and last name on employees with title "sales rep".
  - C. Find name and description on motor cycle models that cost between 50 and 100 \$.
  - D. Find customer numbers for cancelled orders
  - E. Find product line and product vendor of models in product scale 1:18

# Save queries to a file



# Order by

```
SELECT *  
FROM products  
WHERE buyPrice < 25  
ORDER BY productLine;
```

productName	productLine	productScale	productLine
1958 Chevy Corvette Limited Edition	Classic Cars	1:24	C
1982 Lamborghini Diablo	Classic Cars	1:24	S
1936 Harley Davidson El Knucklehead	Motorcycles	1:18	V
1982 Ducati 996 R	Motorcycles	1:32	C
1926 Ford Fire Engine	Trucks and Buses	1:18	C
1936 Mercedes-Benz 500K Special Roadster	Vintage Cars	1:18	S
1939 Cadillac Limousine	Vintage Cars	1:18	S
1939 Chevrolet Deluxe Coupe	Vintage Cars	1:24	M
1938 Cadillac V-16 Presidential Limousine	Vintage Cars	1:24	C
1936 Mercedes Benz 500k Roadster	Vintage Cars	1:24	F

1. List motorcycles sorted by product name.
2. List models with less than 1000 in stock. Sort by quantity in stock with highest quantity in top of the list.
3. List the Norwegian customers' customer name and contact person sorted by primarily their credit limit, secondarily customer name (the latter in descending order).

# Aggregate functions

```
SELECT sum(quantityInStock)
FROM products
WHERE productLine= 'motorcycles';
```

SQL aggregate functions return a single value, calculated from values in a column:

AVG() - Returns the average value

COUNT() - Returns the number of rows

COUNT(DISTINCT )

MAX() - Returns the largest value

MIN() - Returns the smallest value



SUM() - Returns the sum

1. How many german customers are there?
2. What is the average price for classic car models?
3. What is the price of the most expensive model from 'Autoart Studio Design'?
4. How many countries do the customers come from?
5. What is the quantity in stock for 1:12 models?
6. The Products table has a column named "MSRP" (manufacturer's suggested retail price). The column "buyPrice" is "indkøbsprisen". What is the highest profit amongst the products?



# Group by

```
SELECT country, count(customerNumber)
FROM customers
GROUP BY country;
```

Result Grid		  Filter Rows:
	country	count(customerNumb...
▶	Australia	5
	Austria	2
	Belgium	2
	Canada	3
	Denmark	2
	Finland	3
	France	12
	Germany	13
	Hong Kong	1

1. What is the average price for each product line
2. How many different products does each vendor have?
3. What is the profit percentage wise based on product scale?
4. How many orders for each order status type?
5. How many orders do each customer have (customer can be shown by customer number)?
6. Extra: List name of customer with highest credit limit in each country. Order alphabetically by country name.
7. Extra: List the total quantity in stock per product scale, but only if quantity is more than 1000

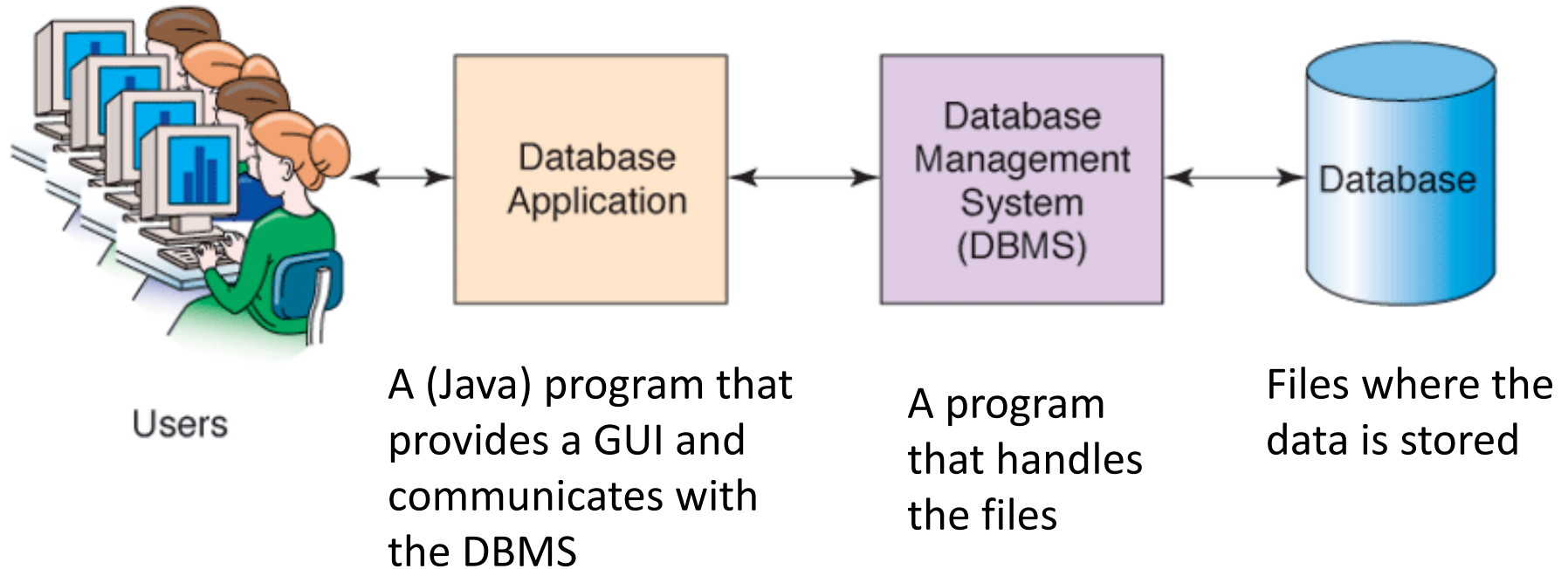


# Database management systems (DBMS)

# The DBMS

A **software system** that enables users to define, create, and maintain the database and that provides controlled access to this database.

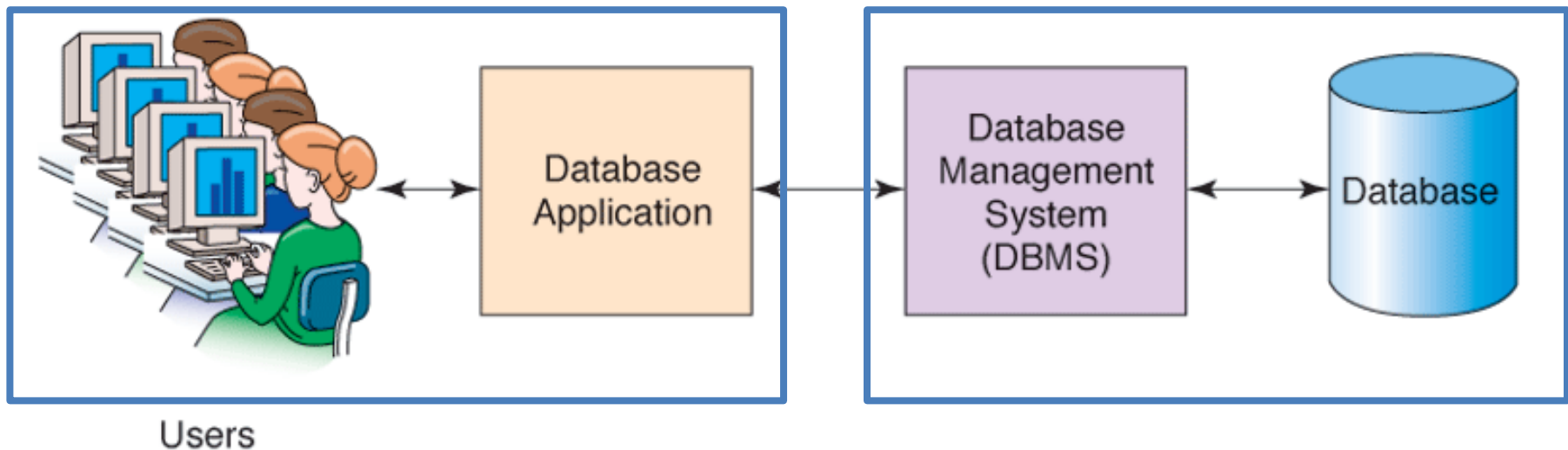
**Figure 1-15** Components of a Database System



# Database and app on each their machine

A **software system** that enables users to define, create, and maintain the database and that provides controlled access to this database.

**Figure 1-15** Components of a Database System

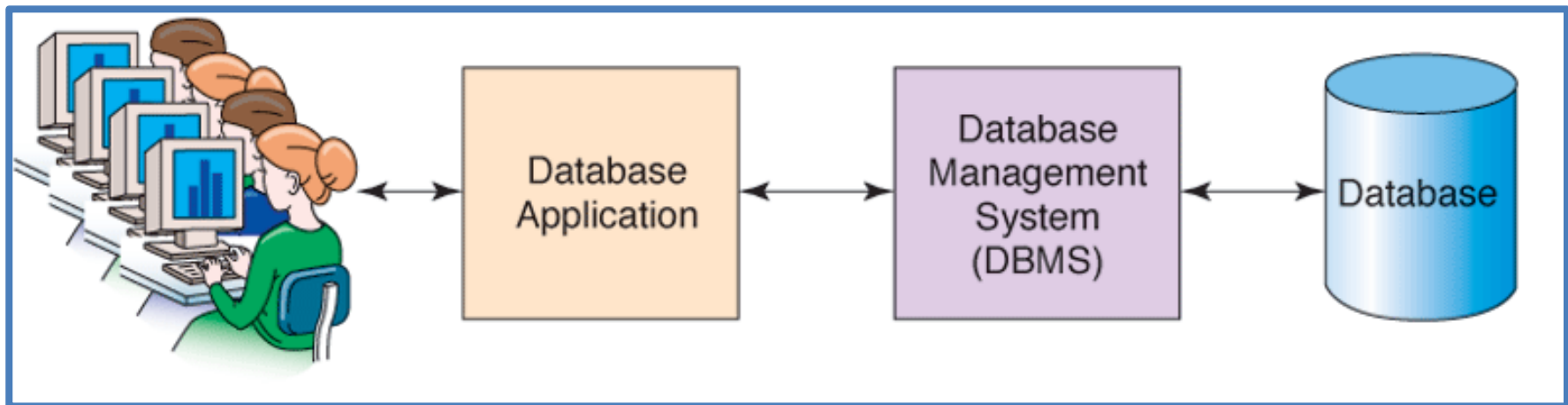


**Called Client-server architecture**

# Database and app on same machine

A **software system** that enables users to define, create, and maintain the database and that provides controlled access to this database.

**Figure 1-15** Components of a Database System



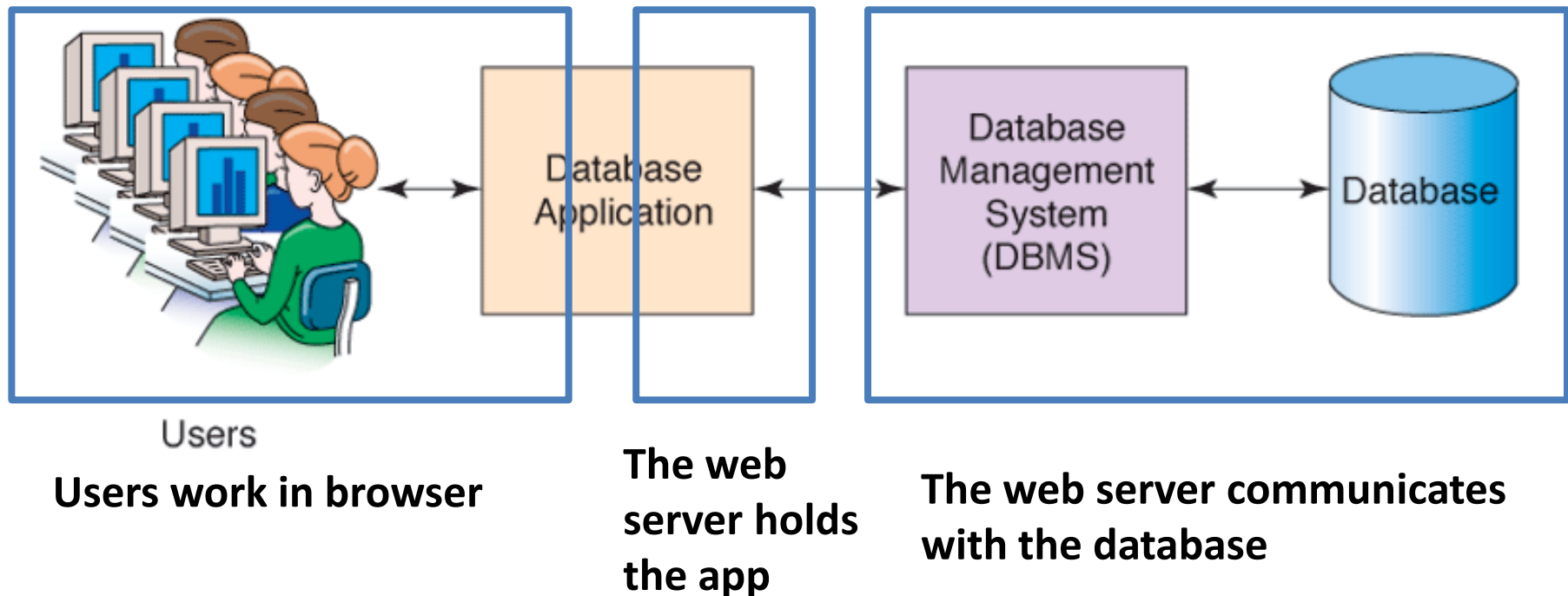
Users

**During development it is more efficient with local database on developer's computer**

# Web based architecture

A **software system** that enables users to define, create, and maintain the database and that provides controlled access to this database.

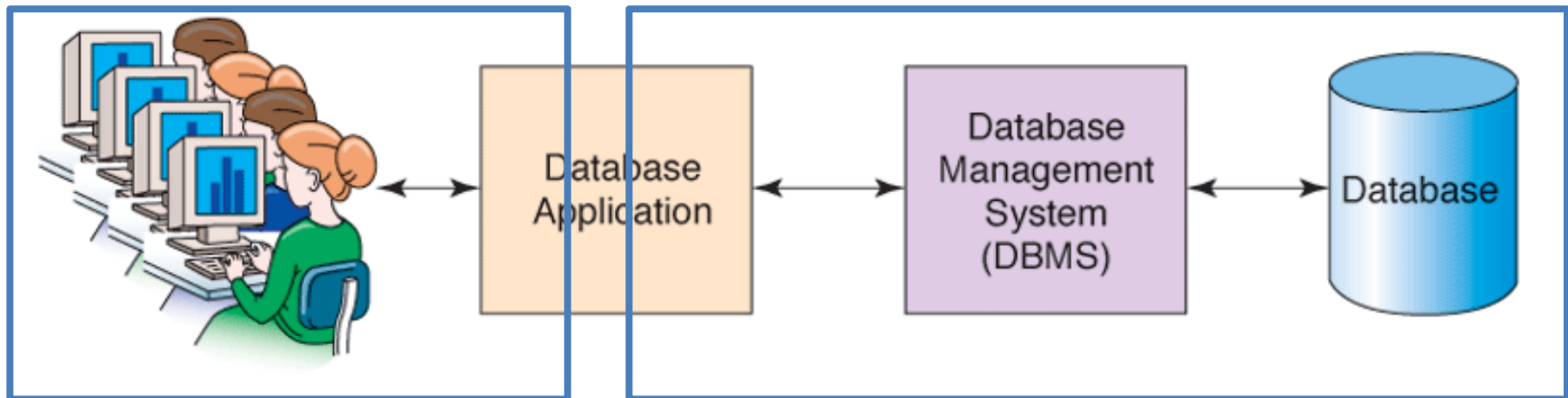
**Figure 1-15** Components of a Database System



# Web based architecture

A **software system** that enables users to define, create, and maintain the database and that provides controlled access to this database.

**Figure 1-15** Components of a Database System



Users

**Users work in browser**

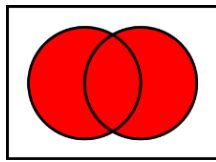
**The web server holds the app, and runs on the same machine as the database**

# Select from more than one table

```
SELECT firstName, lastName, city, addressLine1  
FROM   employees, offices  
WHERE  employees.officeCode = offices.officeCode;
```



# UNION



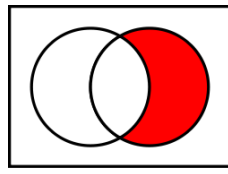
- The union operation of two relational tables is formed by appending rows from one table to those of a second table to produce a third.
- Duplicate rows are eliminated.
- The notation for the union of A and B is A UNION B.
- Tables used in the union operation must be union compatible.
  - Tables that are union compatible must have the same number of columns and corresponding columns must come from the same domain.

A		
k	x	y
1	A	2
2	B	4
3	C	6

B		
k	x	y
1	A	2
4	D	8
5	E	10

A UNION B		
k	x	y
1	A	2
2	B	4
3	C	6
4	D	8
5	E	10

# Difference



- The difference of two relational tables is a third that contains those rows that occur in the first table but not in the second.
- The Difference operation requires that the tables be union compatible.
- As with arithmetic, the order of subtraction matters.

**A**

k	x	y
1	A	2
2	B	4
3	C	6

**B**

k	x	y
1	A	2
4	D	8
5	E	10

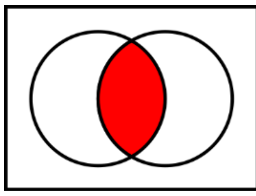
**A - B**

k	x	y
2	B	4
3	C	6

**B - A**

k	x	y
4	D	8
5	E	10

# Intersection



- The intersection of two relational tables is a third table that contains common rows.
- Both tables must be union compatible.
- The notation for the intersection of A and B is A INTERSECT B

A		
k	x	y
1	A	2
2	B	4
3	C	6

B		
k	x	y
1	A	2
4	D	8
5	E	10

A INTERSECT B		
k	x	y
1	A	2

# Product (Cartesian Product)

- The product of two relational tables, also called the Cartesian Product, is the concatenation of every row in one table with every row in the second.
- The product is denoted as A X B

**A**

k	x	y
1	A	2
2	B	4
3	C	6

**B**

k	x	y
1	A	2
4	D	8
5	E	10

**A TIMES B**

ak	ax	ay	bk	bx	by
1	A	2	1	A	2
1	A	2	4	D	8
1	A	2	5	E	10
2	B	4	1	A	2
2	B	4	4	D	8
2	B	4	5	E	10
3	C	6	1	A	2
3	C	6	4	D	8
3	C	6	5	E	10

# Select data from more than one table

Select

<column<sub>i</sub>>, . . . ,

<column<sub>j</sub>>

from

<table<sub>1</sub>> , . . . ,

<table<sub>n</sub>>

[where <condition>]

# Joining Relations

Example from emp-dept database

- In the table for employees called EMP only the numbers of the departments are stored, not the department names.
- For each salesman, we want to retrieve the employee name as well as the name of the department where he is working:

```
select ENAME, E.DEPTNO, DNAME
from emp E, dept D
where E.DEPTNO = D.DEPTNO
and JOB = 'SALESMAN' ;
```

# Example – different ways of writing multi-table queries

deptno	dname	loc
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

```
select ENAME, DNAME
from emp E, dept D
where E.DEPTNO = D.DEPTNO
and JOB = 'SALESMAN' ;
```

empno	ename	job	mgr	hiredate	Sal	Deptno
7369	SMITH	CLERK	7902	12/17/1980	800	20
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	30
7521	WARD	SALESMAN	7698	02/22/1981	1250	30
7566	JONES	MANAGER	7839	04-02-1981	2975	20
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	30
7698	BLAKE	MANAGER	7839	05-01-1981	2850	30
7782	CLARK	MANAGER	7839	06-09-1981	2450	10
7788	SCOTT	ANALYST	7566	04/19/1987	3000	20
7839	KING	PRESIDENT		11/17/1981	5000	10
7844	TURNER	SALESMAN	7698	09-08-1981	1500	30
7876	ADAMS	CLERK	7788	05/23/1987	1100	20
7900	JAMES	CLERK	7698	12-03-1981	950	30
7902	FORD	ANALYST	7566	12-03-1981	3000	20
7934	MILLER	CLERK	7782	01/23/1982	1300	10

```
select ENAME, DNAME
from emp join dept
on emp.deptno = dept.deptno
where job = 'SALESMAN'
```

```
select ENAME, DNAME
from emp
natural join dept
where job = 'SALESMAN'
```

# Subquery

- Select data from one table based on a condition that involves more tables

Example:

Retrieve all room types in Oslo



# Subquery

(Retrieve all room types in Oslo)

```
SELECT type
FROM room
WHERE hotelNo IN (SELECT hotelNo
FROM hotel
WHERE city = 'Oslo');
```

type
dobbelt
suite



Execution order ?

Hotel

<u>hotelNo</u>	hotelName	city
1	SAS	London
2	Hilton	Oslo
3	Grand	Paris
4	Palads	Odense

Room

<u>roomNo</u>	<u>hotelNo</u>	type	price
100	1	dobbelt	600
200	1	dobbelt	700
300	1	enkelt	500
100	2	dobbelt	800
200	2	suite	1200
200	3	dobbelt	900

# Scalar Subquery

```
SELECT type, price - (SELECT MIN(price) FROM room)
      as merpris
FROM room
WHERE hotelNo IN (SELECT hotelNo
                  FROM   hotel
                  WHERE  city = 'London');
```

type	merpris
dobbelt	100
dobbelt	200
enkelt	0



Hotel

<u>hotelNo</u>	hotelName	city
1	SAS	London
2	Hilton	Oslo
3	Grand	Paris
4	Palads	Odense

Room

<u>roomNo</u>	<u>hotelNo</u>	type	price
100	1	dobbelt	600
200	1	dobbelt	700
300	1	enkelt	500
100	2	dobbelt	800
200	2	suite	1200
200	3	dobbelt	900

# Join

Select data from two or more tables

Example:

Retrieve names of all hotels and the room types that the offer

# Join

(Retrieve names of all hotels  
and the room types that the offer)

```
SELECT hotelName, type
FROM   hotel, room
WHERE hotel.hotelNo = room.hotelNo
```

<u>hotelName</u>	type
SAS	dobbelt
SAS	dobbelt
SAS	enkelt
Hilton	dobbelt
Hilton	suite
Grand	dobbelt



Hotel

<u>hotelNo</u>	hotelName	city
1	SAS	London
2	Hilton	Oslo
3	Grand	Paris
4	Palads	Odense

Room

<u>roomNo</u>	<u>hotelNo</u>	type	price
100	1	dobbelt	600
200	1	dobbelt	700
300	1	enkelt	500
100	2	dobbelt	800
200	2	suite	1200
200	3	dobbelt	900

# JOIN



hotelNo	hotelName	city	roomNo	hotelNo	type	price
1	SAS	London	100	1	dobbelt	600
1	SAS	London	200	1	dobbelt	700
1	SAS	London	300	1	enkelt	500
1	SAS	London	100	2	dobbelt	800
1	SAS	London	200	2	suite	1200
1	SAS	London	200	3	dobbelt	900
2	Hilton	Oslo	100	1	dobbelt	600
2	Hilton	Oslo	200	1	dobbelt	700
2	Hilton	Oslo	300	1	enkelt	500
2	Hilton	Oslo	100	2	dobbelt	800
2	Hilton	Oslo	200	2	suite	1200
2	Hilton	Oslo	200	3	dobbelt	900
3	Grand	Paris	100	1	dobbelt	600
3	Grand	Paris	200	1	dobbelt	700
3	Grand	Paris	300	1	enkelt	500
3	Grand	Paris	100	2	dobbelt	800
3	Grand	Paris	200	2	suite	1200
3	Grand	Paris	200	3	dobbelt	900
4	Palads	Odense	100	1	dobbelt	600
4	Palads	Odense	200	1	dobbelt	700
4	Palads	Odense	300	1	enkelt	500
4	Palads	Odense	100	2	dobbelt	800
4	Palads	Odense	200	2	suite	1200
4	Palads	Odense	200	3	dobbelt	900



About execution of the instruction:

```
3 SELECT hotelName, type
1 FROM   hotel, room
2 WHERE  hotel.hotelNo = room.hotelNo
```

1. Creates the **cartesian product**

2. Selects rows on basis of condition

3. Projects columns on basis of attribute names

# Extra things that are nice to know



# Distinct

- Consider the query which shows the room type for each room

```
select TYPE from ROOM;
```

- Duplicate result rows are not automatically eliminated, but use of the keyword `distinct` forces the elimination of duplicates from the query result

```
select distinct TYPE from ROOM;
```

# Arithmetic Operators and Functions

- numbers:  
abs, cos, sin, exp, log, power, mod, sqrt, +, -, \*, /, . . .
- strings:  
concat(string1, string2), lower, upper,  
replace(string, search\_string, replacement\_string),  
substr(string, m, n), length, . . .



# Comparison Operators

- =, != or <>, <, >, <=, =>
- **Set Conditions:** <column> [not] in (<list of values>), e.g.:  
`select * from dept where DEPTNO in (20,30);`
- **Null value:** <column> is [not] null,
  - i.e., for a tuple to be selected there must (not) exist a defined value for this column. E.g.:  
`select * from emp where MGR is not null;`
- **Domain conditions:** <column> [not] between <lower bound> and <upper bound>. E.g.:

```
select EMPNO, ENAME, SAL from emp
where SAL between 1500 and 2500;
```

```
select ENAME from emp
where HIREDATE between '1981-04-02' and
'1981-09-08';
```

# More on String Operations

- **upper(<string>)** takes a string and converts any letters in it to uppercase, e.g.,  

```
DNAME = upper(DNAME)
```
- **lower(<string>)** converts any letter to lowercase,
- **length(<string>)** returns the length of the string.
- **substr(<string>, n [, m])** clips out a m character piece of <string>, starting at position n. If m is not specified, the end of the string is assumed.

```
substr('DATABASE SYSTEMS', 10, 7)
```

returns the string 'SYSTEMS'.