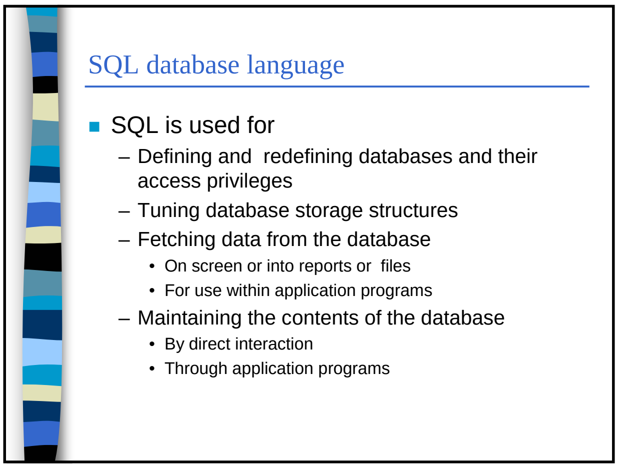
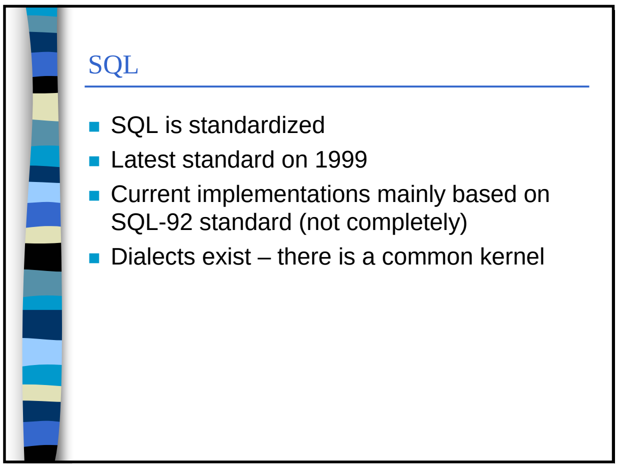


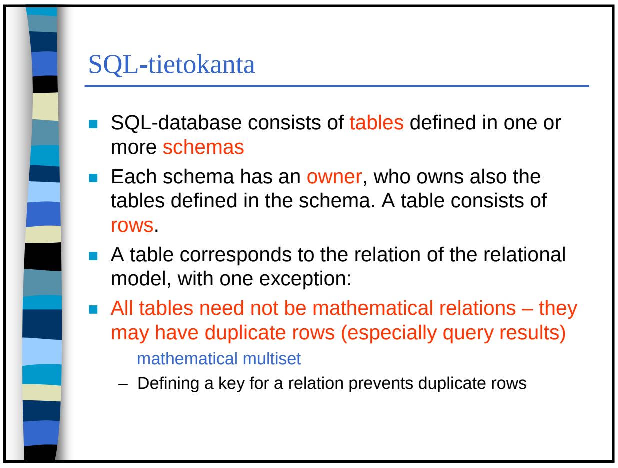
SQL database language

- SQL is used for
  - Defining and redefining databases and their access privileges
  - Tuning database storage structures
  - Fetching data from the database
    - On screen or into reports or files
    - For use within application programs
  - Maintaining the contents of the database
    - By direct interaction
    - Through application programs



SQL

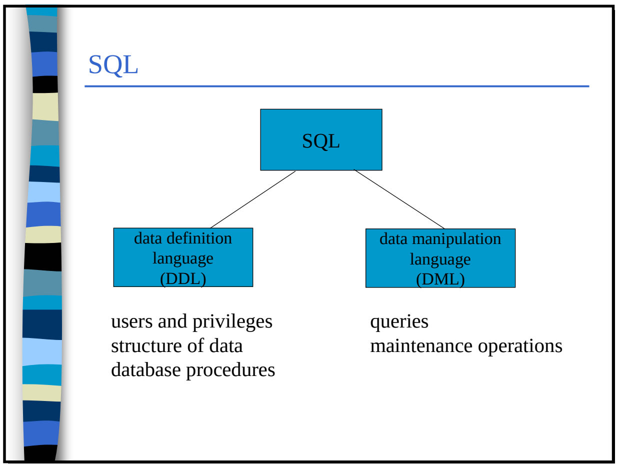
- SQL is standardized
- Latest standard on 1999
- Current implementations mainly based on SQL-92 standard (not completely)
- Dialects exist – there is a common kernel



SQL-tietokanta

- SQL-database consists of tables defined in one or more schemas
- Each schema has an owner, who owns also the tables defined in the schema. A table consists of rows.
- A table corresponds to the relation of the relational model, with one exception:
- All tables need not be mathematical relations – they may have duplicate rows (especially query results)

  mathematical multiset
  - Defining a key for a relation prevents duplicate rows

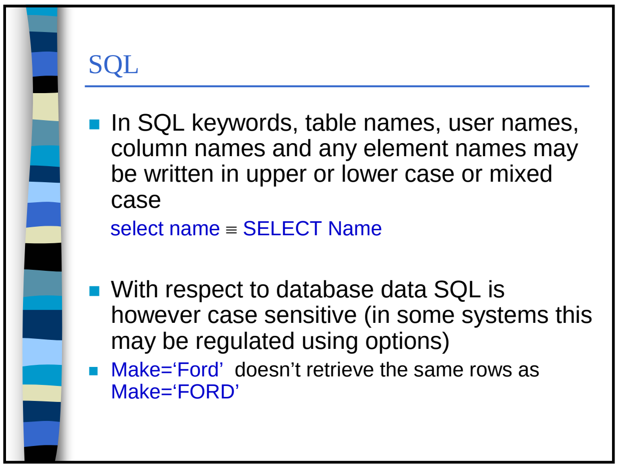


SQL

SQL
data definition language (DDL)
data manipulation language (DML)

users and privileges
structure of data
database procedures

queries
maintenance operations

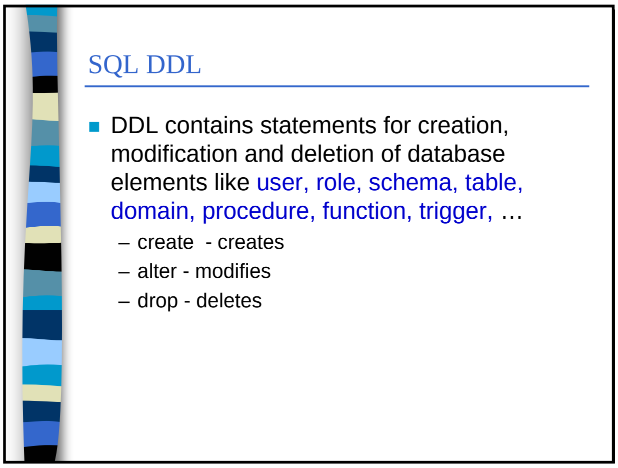


SQL

- In SQL keywords, table names, user names, column names and any element names may be written in upper or lower case or mixed case

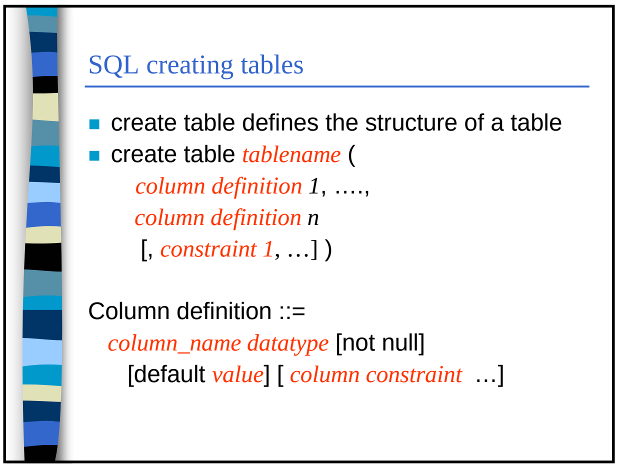
  select name ≡ SELECT Name

- With respect to database data SQL is however case sensitive (in some systems this may be regulated using options)
- Make='Ford' doesn't retrieve the same rows as Make='FORD'



SQL DDL

- DDL contains statements for creation, modification and deletion of database elements like user, role, schema, table, domain, procedure, function, trigger, …
  - create - creates
  - alter - modifies
  - drop - deletes



SQL creating tables

- create table defines the structure of a table
- create table *tablename* (
  column definition 1, ….,
  column definition n
  [, *constraint 1*, …])

Column definition ::=
column_name datatype [not null]
[default *value*] [*column constraint* …]

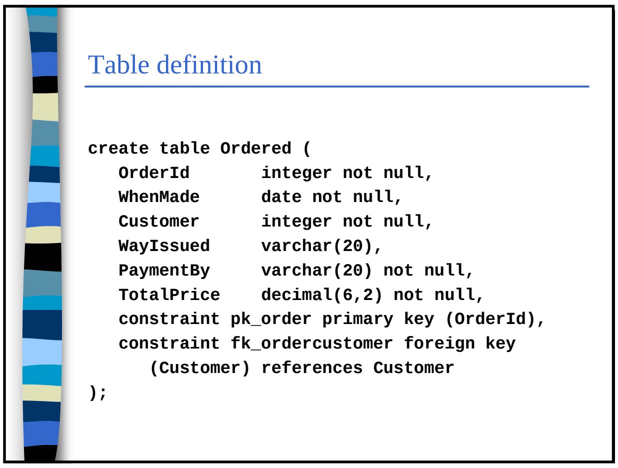


Table definition

```
create table Ordered (
   OrderId       integer not null,
   WhenMade      date not null,
   Customer      integer not null,
   WayIssued     varchar(20),
   PaymentBy     varchar(20) not null,
   TotalPrice    decimal(6,2) not null,
   constraint pk_order primary key (OrderId),
   constraint fk_ordercustomer foreign key
      (Customer) references Customer
);
```

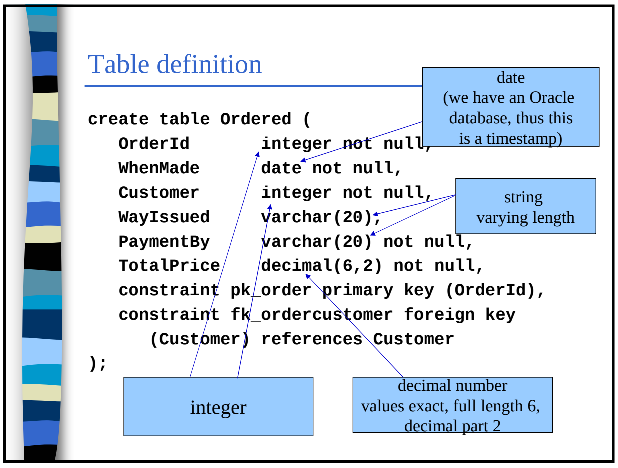


Table definition

```
create table Ordered (
   OrderId       integer not null,
   WhenMade      date not null,
   Customer      integer not null,
   WayIssued     varchar(20),
   PaymentBy     varchar(20) not null,
   TotalPrice    decimal(6,2) not null,
   constraint pk_order primary key (OrderId),
   constraint fk_ordercustomer foreign key
      (Customer) references Customer
);
```

date (we have an Oracle database, thus this is a timestamp)

string varying length

integer

decimal number values exact, full length 6, decimal part 2

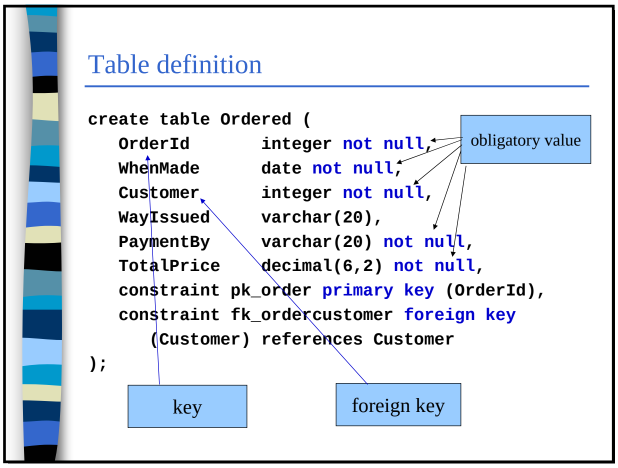


Table definition

```
create table Ordered (
   OrderId       integer not null,
   WhenMade      date not null,
   Customer      integer not null,
   WayIssued     varchar(20),
   PaymentBy     varchar(20) not null,
   TotalPrice    decimal(6,2) not null,
   constraint pk_order primary key (OrderId),
   constraint fk_ordercustomer foreign key
      (Customer) references Customer
);
```

obligatory value

key

foreign key

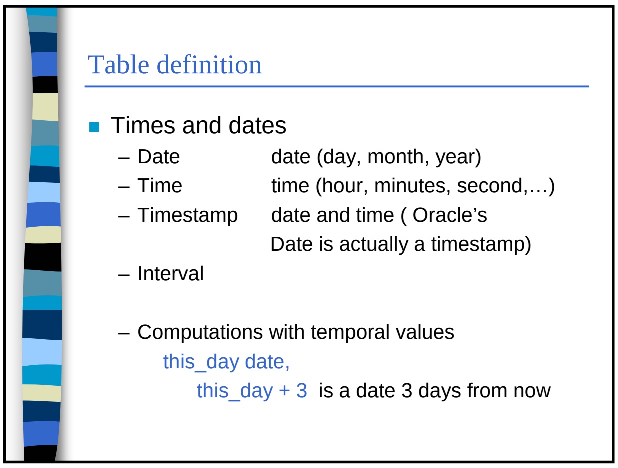


Table definition

- Times and dates
  - Date date (day, month, year)
  - Time time (hour, minutes, second,…)
  - Timestamp date and time (Oracle's Date is actually a timestamp)
  - Interval
  - Computations with temporal values
    this_day date,
    this_day + 3 is a date 3 days from now

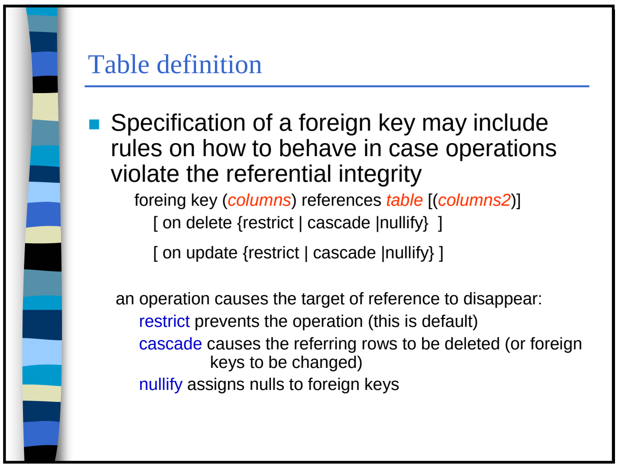
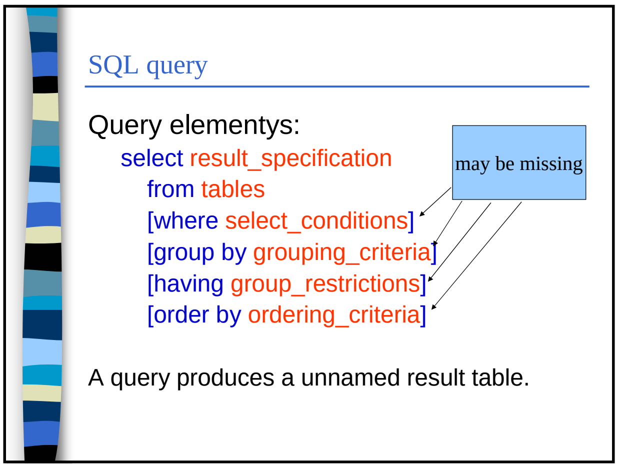


Table definition

- Specification of a foreign key may include rules on how to behave in case operations violate the referential integrity

foreing key (*columns*) references *table* [(*columns2*)]
[on delete {restrict | cascade |nullify}]
[on update {restrict | cascade |nullify}]

an operation causes the target of reference to disappear:
restrict prevents the operation (this is default)
cascade causes the referring rows to be deleted (or foreign keys to be changed)
nullify assigns nulls to foreign keys



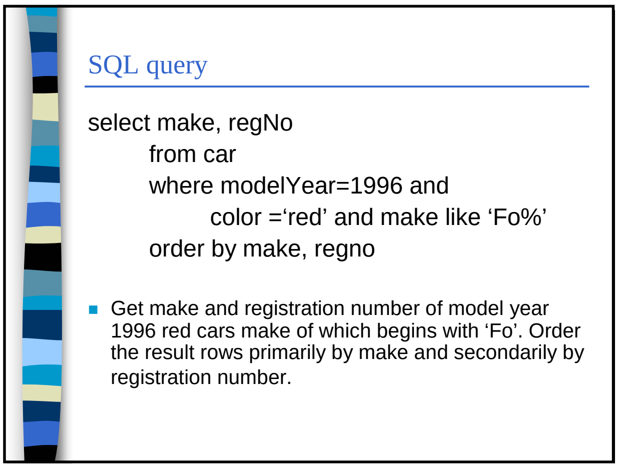
SQL query

Query elementys:

```
    select result_specification
       from tables
       [where select_conditions]
       [group by grouping_criteria]
       [having group_restrictions]
       [order by ordering_criteria]
```

may be missing

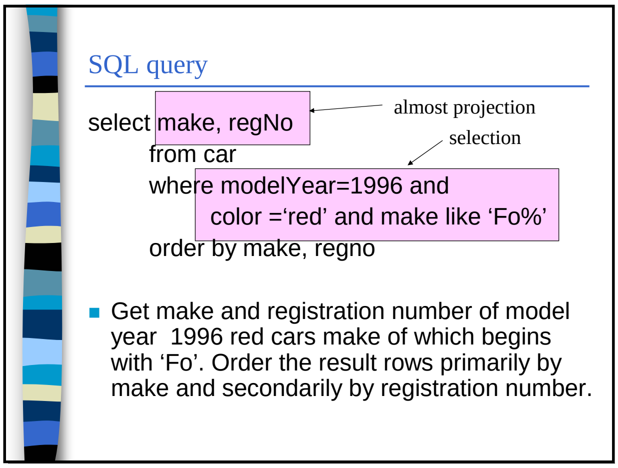
A query produces a unnamed result table.



SQL query

```
select make, regNo
      from car
      where modelYear=1996 and
              color ='red' and make like 'Fo%'
      order by make, regno
```

- Get make and registration number of model year 1996 red cars make of which begins with ‘Fo’. Order the result rows primarily by make and secondarily by registration number.



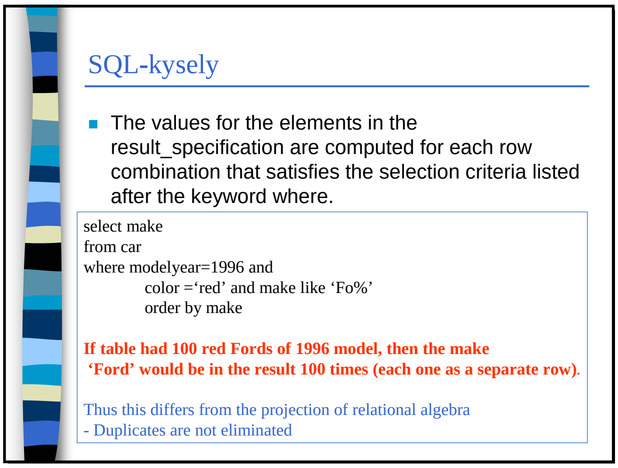
SQL query

```
select make, regNo
       from car
       where modelYear=1996 and
             color ='red' and make like 'Fo%'
       order by make, regno
```

almost projection

selection

- Get make and registration number of model year 1996 red cars make of which begins with ‘Fo’. Order the result rows primarily by make and secondarily by registration number.



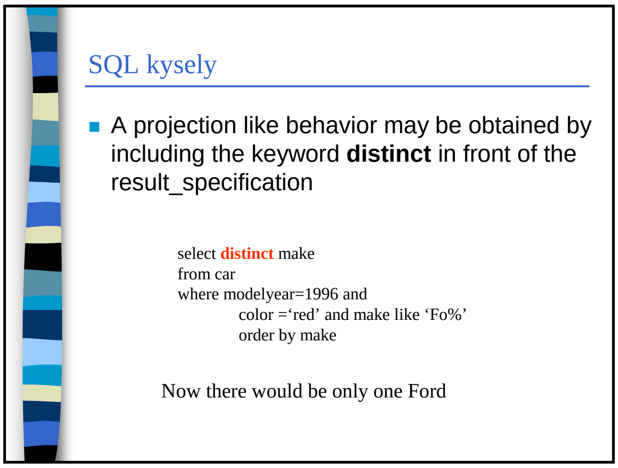
SQL-kysely

- The values for the elements in the result_specification are computed for each row combination that satisfies the selection criteria listed after the keyword where.

```
select make
from car
where modelyear=1996 and
        color ='red' and make like 'Fo%'
        order by make
```

If table had 100 red Fords of 1996 model, then the make ‘Ford’ would be in the result 100 times (each one as a separate row).

Thus this differs from the projection of relational algebra
- Duplicates are not eliminated

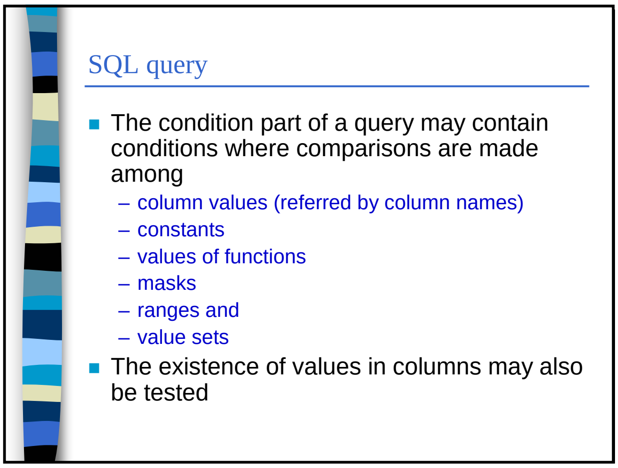


SQL kysely

- A projection like behavior may be obtained by including the keyword **distinct** in front of the result_specification

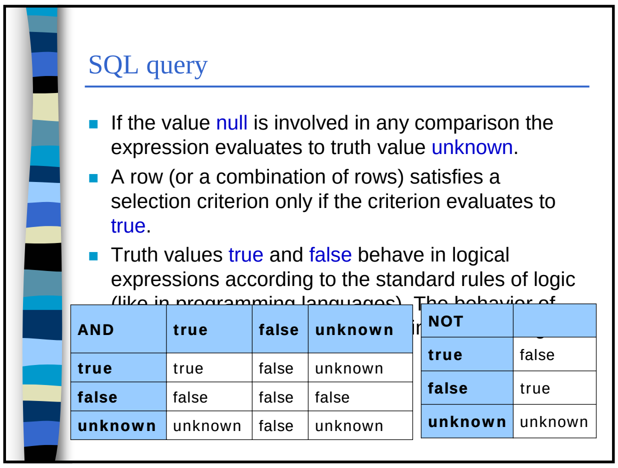
```
select distinct make
from car
where modelyear=1996 and
        color ='red' and make like 'Fo%'
        order by make
```

Now there would be only one Ford



SQL query

- The condition part of a query may contain conditions where comparisons are made among
  - column values (referred by column names)
  - constants
  - values of functions
  - masks
  - ranges and
  - value sets
- The existence of values in columns may also be tested

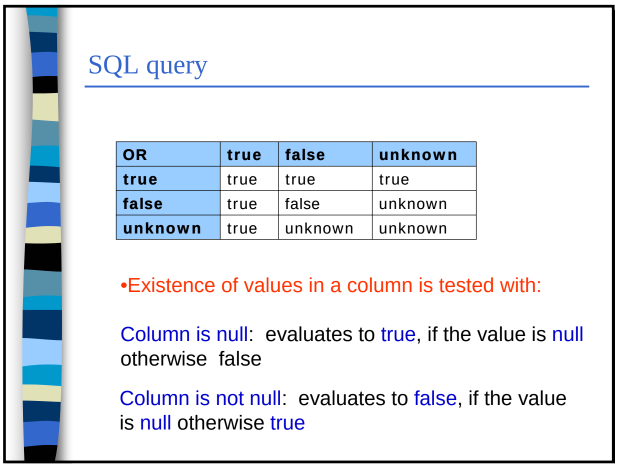


SQL query

- If the value null is involved in any comparison the expression evaluates to truth value unknown.
- A row (or a combination of rows) satisfies a selection criterion only if the criterion evaluates to true.
- Truth values true and false behave in logical expressions according to the standard rules of logic (like in programming languages). The behavior of

AND	true	false	unknown
true	true	false	unknown
false	false	false	false
unknown	unknown	false	unknown

NOT	
true	false
false	true
unknown	unknown



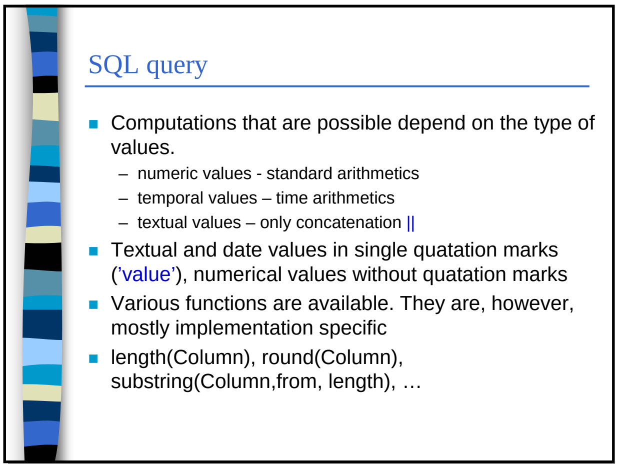
SQL query

OR	true	false	unknown
true	true	true	true
false	true	false	unknown
unknown	true	unknown	unknown

•Existence of values in a column is tested with:

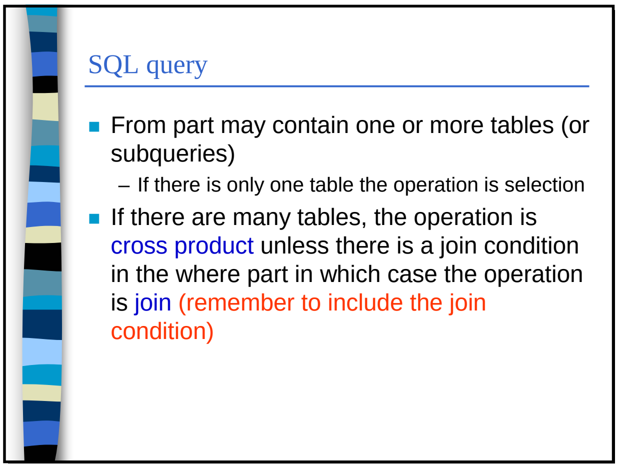
Column is null: evaluates to true, if the value is null otherwise false

Column is not null: evaluates to false, if the value is null otherwise true



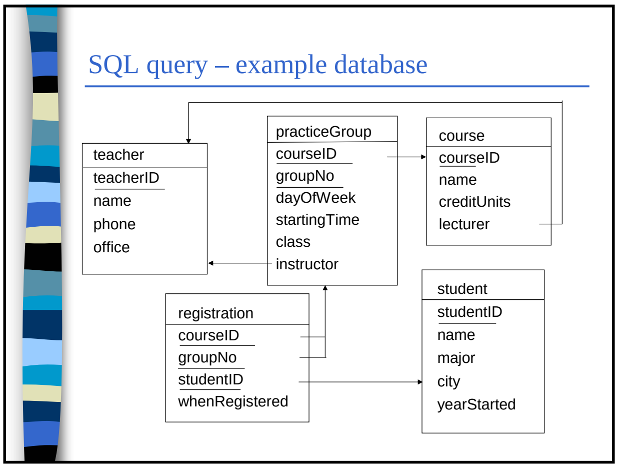
SQL query

- Computations that are possible depend on the type of values.
  - numeric values - standard arithmetics
  - temporal values – time arithmetics
  - textual values – only concatenation ||
- Textual and date values in single quatation marks ('value'), numerical values without quatation marks
- Various functions are available. They are, however, mostly implementation specific
- length(Column), round(Column), substring(Column,from, length), …



SQL query

- From part may contain one or more tables (or subqueries)
  - If there is only one table the operation is selection
- If there are many tables, the operation is cross product unless there is a join condition in the where part in which case the operation is join (remember to include the join condition)



SQL query – example database

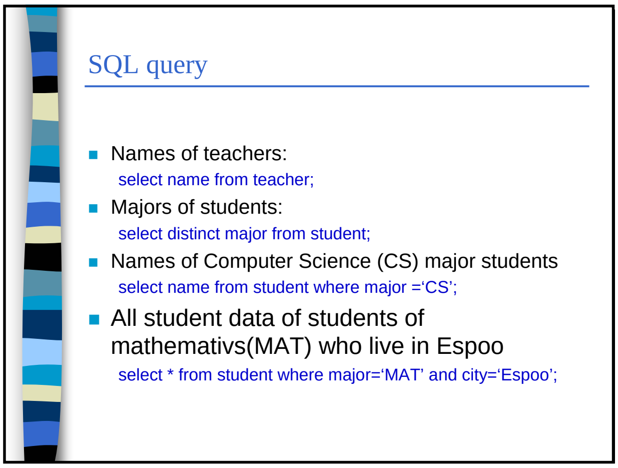
teacher
teacherID
name
phone
office

practiceGroup
courseID
groupNo
dayOfWeek
startingTime
class
instructor

course
courseID
name
creditUnits
lecturer

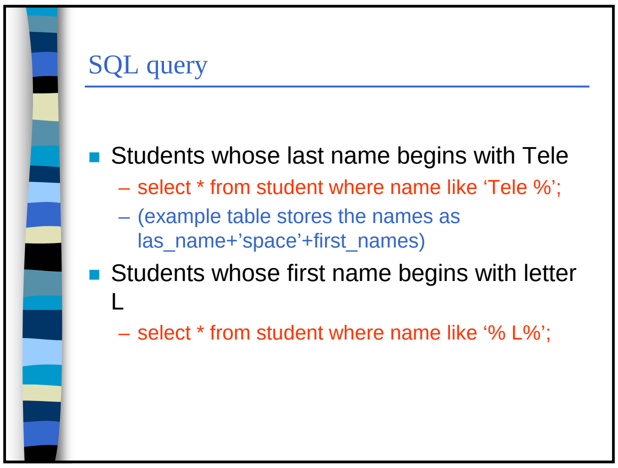
registration
courseID
groupNo
studentID
whenRegistered

student
studentID
name
major
city
yearStarted



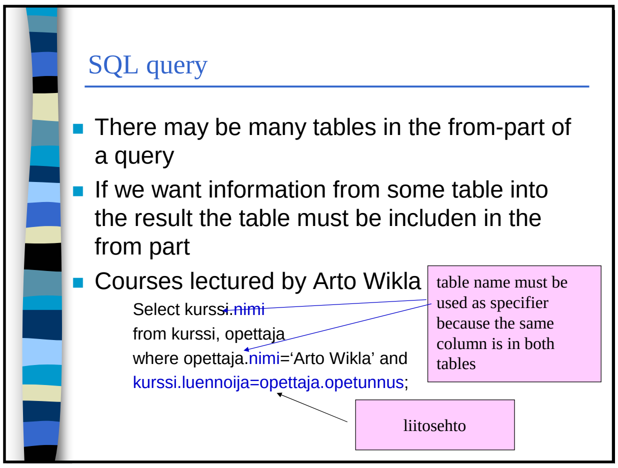
SQL query

- Names of teachers:
  select name from teacher;
- Majors of students:
  select distinct major from student;
- Names of Computer Science (CS) major students
  select name from student where major ='CS';
- All student data of students of mathemativs(MAT) who live in Espoo
  select * from student where major='MAT' and city='Espoo';



SQL query

- Students whose last name begins with Tele
  - select * from student where name like 'Tele %';
  - (example table stores the names as las_name+'space'+first_names)
- Students whose first name begins with letter L
  - select * from student where name like '% L%';



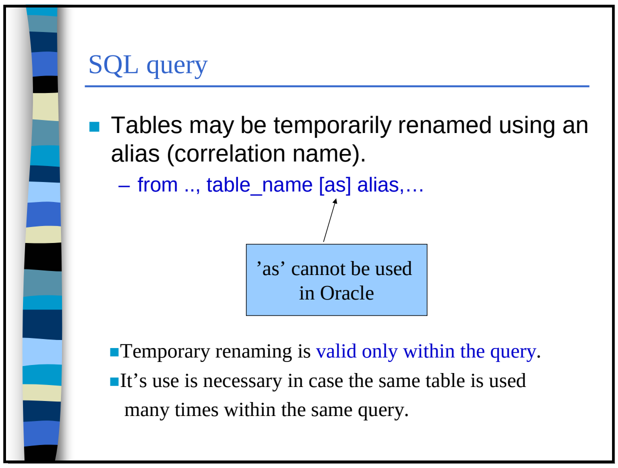
SQL query

- There may be many tables in the from-part of a query
- If we want information from some table into the result the table must be includen in the from part
- Courses lectured by Arto Wikla

```
Select kurssi.nimi
from kurssi, opettaja
where opettaja.nimi='Arto Wikla' and
kurssi.luennoija=opettaja.opetunnus;
```

table name must be used as specifier because the same column is in both tables

liitosehto

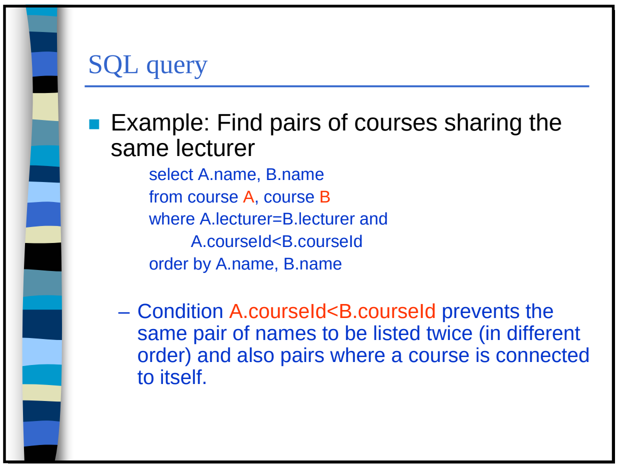


SQL query

- Tables may be temporarily renamed using an alias (correlation name).
  - from .., table_name [as] alias,…

'as' cannot be used in Oracle

- Temporary renaming is valid only within the query.
- It's use is necessary in case the same table is used many times within the same query.

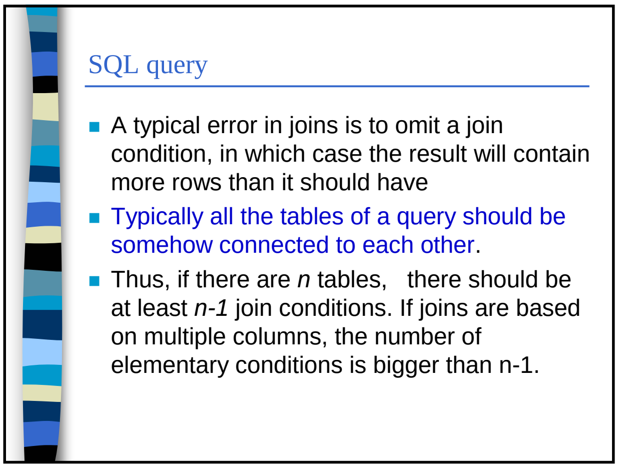


SQL query

- Example: Find pairs of courses sharing the same lecturer

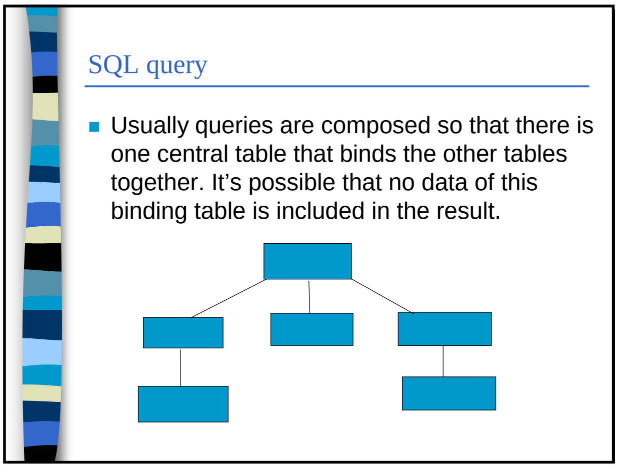
```
select A.name, B.name
from course A, course B
where A.lecturer=B.lecturer and
        A.courseId<B.courseId
order by A.name, B.name
```

  - Condition A.courseId<B.courseId prevents the same pair of names to be listed twice (in different order) and also pairs where a course is connected to itself.



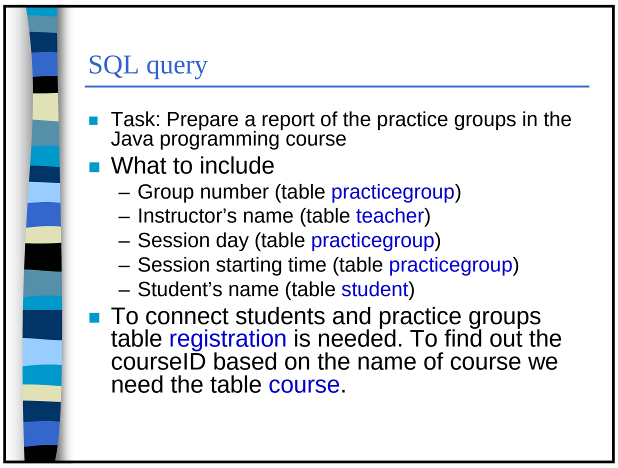
SQL query

- A typical error in joins is to omit a join condition, in which case the result will contain more rows than it should have
- Typically all the tables of a query should be somehow connected to each other.
- Thus, if there are *n* tables, there should be at least *n-1* join conditions. If joins are based on multiple columns, the number of elementary conditions is bigger than n-1.



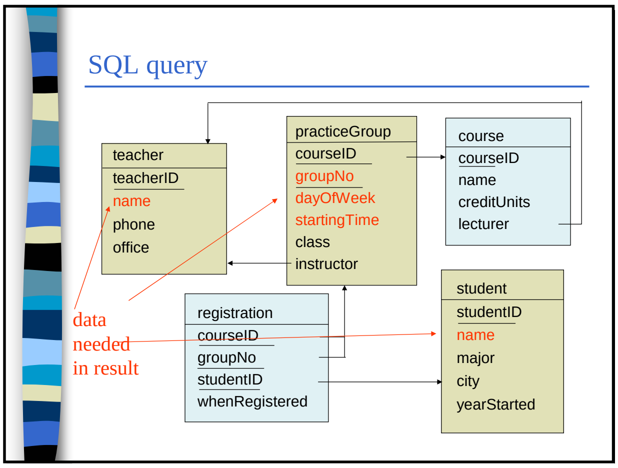
SQL query

- Usually queries are composed so that there is one central table that binds the other tables together. It's possible that no data of this binding table is included in the result.

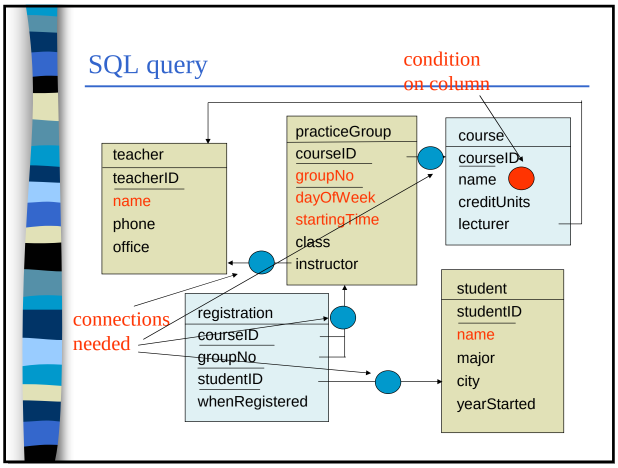


SQL query

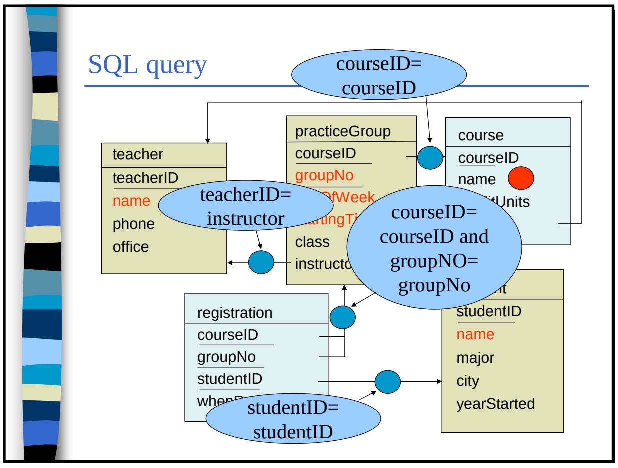
- Task: Prepare a report of the practice groups in the Java programming course
- What to include
  - Group number (table practicegroup)
  - Instructor's name (table teacher)
  - Session day (table practicegroup)
  - Session starting time (table practicegroup)
  - Student's name (table student)
- To connect students and practice groups table registration is needed. To find out the courseID based on the name of course we need the table course.



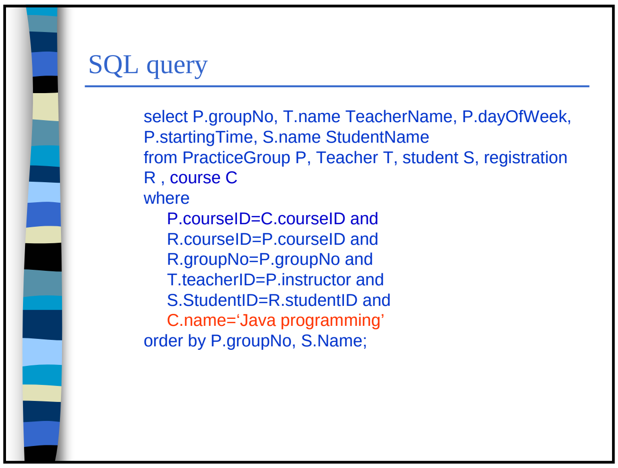
SQL query



SQL query



SQL query



SQL query

```
select P.groupNo, T.name TeacherName, P.dayOfWeek,
P.startingTime, S.name StudentName
from PracticeGroup P, Teacher T, student S, registration
R , course C
where
    P.courseID=C.courseID and
    R.courseID=P.courseID and
    R.groupNo=P.groupNo and
    T.teacherID=P.instructor and
    S.StudentID=R.studentID and
    C.name='Java programming'
order by P.groupNo, S.Name;
```