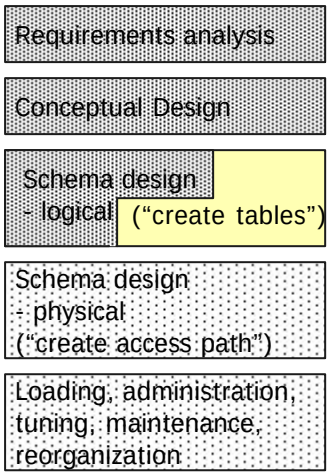
4 Logical Design : RDM
Schema Definition with SQL / DDL

4.1 SQL history and standards
4.2 SQL/DDL – first steps
4.2.1 Basis Schema Definition using SQL / DDL
4.2.2 SQL Data types, domains, user defined types
4.2.3 Creating simple tables

see: Melton/Simon: chap 2, 3.3, 4
System documentation (e.g. Postgres, Oracle, MySQL, see references)

Logical Design Lifecycle

- Context



System analyst
DB designer
Application programmer

Application programmer
DB administrator

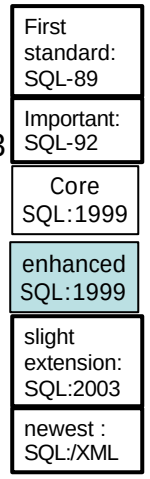
DB administrator

4.1 SQL History

1974	Prototype “System R” (IBM, San Jose) First relational DBMS based on Codd’s relational model Structured English Query Language (SEQUEL)
1975	SEQUEL renamed SQL (pronounced “Sequel” in US)
1986	First standardization attempt based on system R
1989	SQL standard ANSI SQL-1 , SQL-89 about 120 pages
1992	SQL2 standard SQL : *Standard* Query Language ANSI SQL-2, SQL-92 about 600 pages

Standard Query Language: Standards

- SQL-92 compliance levels:
  (1) Entry SQL: basically SQL-89, essential
  (2) Intermediate SQL,
  (3) Full SQL
  – No implementation of SQL-92 on level 2 or3
    - SQL 1999 (SQL3) levels:
      - Core SQL: essential for standard compliance
      - Additional Features, e.g. object features



- Standards: not “nice to have” but inevitable
- Heavy influenced by strategies of SW-Industry
- All known implementations do not conform to every aspect of the standard

Query Language Course Information

Within the course:

- Basic concepts of SQL:1999
- Oracle9i Core SQL:1999 compliant + additional features
- [MySQL 3.x not SQL compliant (no subqueries, foreign keys,…), more features in Vers. 4.x]
- Postgres: close to SQL99 like Oracle

- Self study of further SQL concepts
- Local Oracle-documentation:
  http://www.inf.fu-berlin.de/inst/ag-db/software/oracle/oracle_docu/index.htm
- Postgres: help files – very technical and detailled
- SQL3 Syntax : local web pages

SQL Different sublanuages

- Data definition Language (DDL)
  - Definition and change of data structures on all three database levels: Namespaces, relations with attributes, domains, data types, integrity constraints, triggers, functions on database,views, placement of data, space needed, access structures,…
  - Heavily system depend, least common denominator still SQL 92 entry level

- Data manipulation language (DML)
  - Create, change, delete data
  - Interactive query formulation
  - Embedding of SQL commands in host language
  - Specification of begin, abort, and end of transaction

- Data Administration language
  - Access rights, authorization

4.2 SQL / DDL – first steps

4.2.1 Basis Schema Definition using SQL / DDL

Defining the relational schema: concepts to be expressed

Logical Schema

- Namespaces (database, schema, ...)
- Domains, data types
- Tables
- Integrity constraints
  - Key and uniqueness constraints
  - Value constraints
  - Cardinality constraints

Schema definition using SQL/DDL

More…

- Active elements (triggers)
- Functions defined on the database

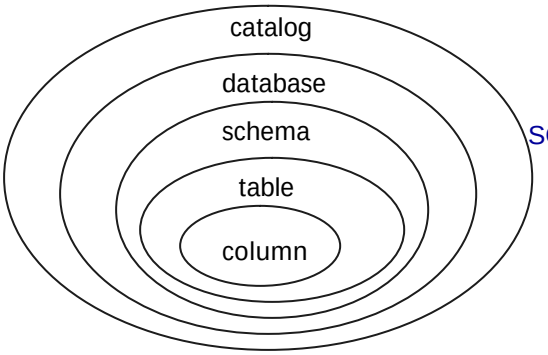
Physical Schema

- Physical Storage
  - Space needed
  - Access structures
  - Physical placement of data

Physical Schema: postponed

Name spaces in SQL / DDL

database-name space? schema name space ?



SQL-2 standard

Name structure:

<cat>.<database>.<schema>.<table>.<column>

Databases and schemas in Postgres

DB cluster

set of databases ≡ catalog / SQL 99

⇩

Database

physically separated set of schemas

⇩

Schema

logical construct; set of "database objects"

⇩

Tables, functions, triggers, ...

Namespace: **database.schema.table**

without effect in Postgres

Schemas in standard SQL / DDL and DBS

CREATE SCHEMA <schemaName>

e.g **CREATE SCHEMA VideoStoreDB**

creates a namespace, in which relations (tables) have unambiguous names

– Has been proposed by SQL-2, but no DBS supports the full naming scheme

– Only **<table>.<column>** names are supported by all systems, confusing terminology in systems

Oracle:

Database = set of physical storage areas ("tablespaces")
Name of schema = dbsUsername , all objects may be prefixed with <dbUsername>

MySQL:

Database = directory in File system where data reside
Schema not defined in MySQL

4.2.2 SQL Data types

- Primitive attribute (column) types
  - Base types of the SQL and/or DB system
  - No constructed types
    contradict „first normal form“ – introduced by SQL99
  - Types for numbers, characters, strings, date / time, Binary objects

Numeric datatypes in SQL-2

- **NUMERIC (p,s)** exact number, basically same
- **DECIMAL (p,s)** as **DECIMAL**
- **INTEGER** alias: **INT**
- **SMALLINT**
- **FLOAT (p,s)** approximate number
- **REAL** implementation dependent precision
- **DOUBLE PRECISION**

SQL Built-in types

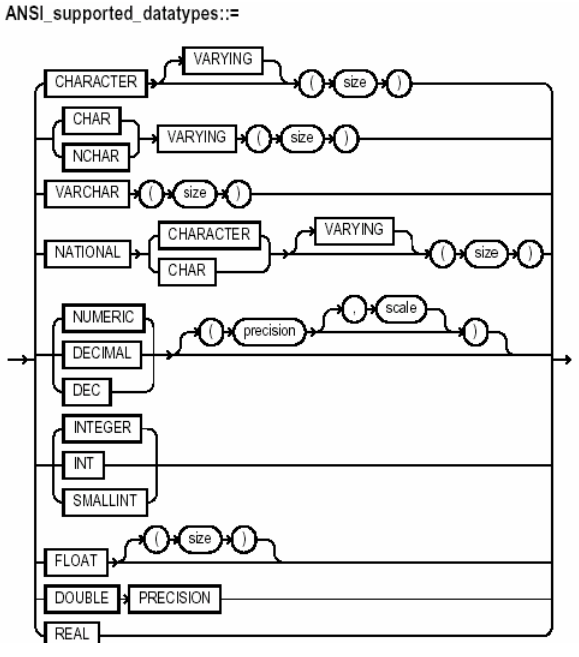
More datatypes in SQL-2: Character etc

	Literal
CHARACTER [(n)] CHAR	**'A padded string '**
// fixed length character string	
CHARACTER VARYING (n)	
VARCHAR (n)	**'Hello SQL'**
// variable length string, n=maximum	

NATIONAL CHARACTER (n) | NCHAR (n)
NCHAR VARYING (n)
not **NVARCHAR (n)** as in SQL99, do not use it in Oracle e.g.

DATE	**DATE '2001-5-2'**
TIME	**TIME '01:00:05.01'**
TIMESTAMP	**composed of year, month, day, hour, minute, second**

INTERVAL FirstUnitofTime [to LastUnitofTime]
e.g. '1 day 12 hours 59 min 10 sec'



Syntax diagram for ANSI / SQL-2 character data type

“Large Objects”

Large Character / Binary Objects since SQL 1999

Restricted, implementation defined restriction of maximum character string length
`Char(n)` / **`VARCHAR(n)`**, typically 4000 Bytes

`CHARACTER LARGE OBJECT`	**`CLOB`**
`NATIONAL CHARACTER LARGE OBJECT`	**`NCLOB`**
`BINARY LARGE OBJECT`	**`BLOB`**

Typically up to 2 GB or even more.

Useful for images, videos, …

No blobs in Postgres ... but 'bytea' binary data type and arbitrary long 'te xt' data type.

Postgres specific data types

- Net specific
  - **`macaddr`** MAC address
  - **`inet`** IPV4 / V6 address
- Geometric types
  - **`point`**
  - **`lseg`** line segment
  - **`path`** closed or open path
  - **`polygon, box`**
  - **`circle`**
- Miscalleneous
  - serial autoincremented 32-Bit-Integer
- Constructed types
  - arrays and more....

Oracle SQL built-in types

ORACLE

Datatype	Description
• **VARCHAR2(*size*)**	Variable-length character data
• **CHAR(*size*)**	Fixed-length character data
• **NUMBER(*p,s*)**	Variable-length numeric data
• **DATE**	Date and time values
• **LONG**	Variable-length character data up to 2 gigabytes
• **CLOB**	Single-byte character data up to 4 gigabytes
• **RAW(n),LONG RAW**	Raw binary data (up to 2 KB | 2 GB)
• **BLOB**	Binary data up to 4 gigabytes e.g. **X'49FE'**
• **BFILE**	Binary data stored in an external file; up to 4 gigabytes

Differences

- Numeric types in different DBS:
  - Oracle

    NUMBER(*p,s*) Variable-length numeric data
  - MySQL:

```
TINYINT[(M)], SMALLINT[(M)], MEDIUMINT[(M)],
INT[(M)], BIGINT[(M)], FLOAT(precision),
FLOAT[(M,D)],  DOUBLE[(M,D)], DOUBLE
PRECISION[(M,D)], REAL[(M,D)],
DECIMAL[(M[,D])], NUMERIC[(M[,D])]
```

- Many differences from standard

> • Always use standard types
> • Makes database less independent from the database system vendor

SQL/DDL Domains

Domain

– named sets of values

CREATE DOMAIN <domainName> <typeDef>

CREATE DOMAIN Money DECIMAL (10,2)

not really representation independent, but useful in order to avoid semantically meaningless operations, e.g. comparing money with length attributes

– Not supported in most Systems (neither Oracle nor MySQL, exception Postgres, SAP-DB)

SQL / DDL: Type definitions (user defined type)

Core SQL:1999

- Distinct type:
  - Similar to domain definition
  - Strong typing
- Syntax:

```
CREATE TYPE <typeName> as <typeDef>
  [FINAL];
```

- Examples:

```
CREATE TYPE Euro AS DECIMAL(8,2) FINAL;
CREATE TYPE Mark AS DECIMAL(8,2) FINAL;
CREATE Type Address AS(
        street    varchar (25),
        zipCode   Integer,....);
```

4.2.3 Creating simple tables

SQL is basically case-insensitive

Tables

```
CREATE TABLE  <TableName> (
  <attributName><attributeType>
  [<constraint>]
{,<attributName><attributeType>
  [<constraint>]}
{,<tableConstraints>});
```

```
CREATE TABLE Format (
  format CHAR(5), extraCh DECIMAL
  (3,2) )
```

Basic definition, many more specification options, even physical placement of tables

SQL / DDL: Table Definition

Simple example

```
CREATE TABLE Movie (
/* The Movie table does only store
   basic facts about the movies
*/
 m_id        INTEGER PRIMARY KEY,
 title       CHAR VARYING(60) NOT NULL,
 cat         CHAR(20),
 year        DATE,
 director    VARCHAR(40),
 price_Day   DECIMAL(4,2),
 length      INTEGER)
```

No constraints up to now except PRIMARY KEY
e.g. multi-attribute primary key, cardinality constraints etc. lost.