ingenieur
wissenschaften
htw saar

Digital Skills für Ingenieur*innen

5. Vorlesungstag

„Datenbanken“

Datenbanken

Physischer Entwurf (Implementierung)

- Festlegung der Datentypen der Attribute ggf. mit weiteren Einschränkungen zur Sicherstellung der Integrität und Konsistenz.
- Festlegung von Sichten und Zugriffsrechten zur Sicherstellung des Datenschutzes und der Datensicherheit
- Implementierung des logischen Schemas im DBMS.
- Einfügen der eventuell schon existierenden Grunddaten.

ingenieur
wissenschaften
htw saar

Datenbanken

Interaktionen mit Datenbanken

- Anlegen einer Datenbank
  CREATE DATABASE datenbankname;
- Anzeigen der Datenbank(en)
  SHOW DATABASES;
- Datenbank auswählen
  USE datenbankname;
- Datenbank löschen
  DROP DATABASE datenbankname;

Interaktionen mit Tabellen

- Anlegen einer Datenbanktabelle

```
CREATE TABLE tabellenname
(
   # Attribute
);
```

- Löschen einer Datenbanktabelle
  DROP TABLE tabellenname;

ingenieur
wissenschaften
htw saar

Datenbanken

Datentypen

- Zur Erstellung der Relationen können verschiedene Datentypen genutzt werden.
- Grob lassen sich vier verschiedene Datentyp Kategorien beschreiben:
  - Numerische Datentypen: TINYINT, BOOLEAN, INT1, SMALLINT, … Die meisten numerischen Datentypen können als SIGNED, UNSIGNED und ZEROFILL definitiert werden.
  - String Datentypen: CHAR, CHAR BYTE, VARCHAR, TEXT, … Diese Datentypen nehmen unterschiedlich lange und verschiedenartige Zeichenketten auf.
  - Daten und Zeitstempel: DATE, TIME, DATETIME, TIMESTAMP … Mit diesen Datentypen werden Zeit- und Jahreszahlen gespeichert.
  - Andere Datentypen: AUTO_INCREMENT, NULL Values, Geometry Types, …

Datenbanken

Numerische Datentypen

- TINYINT[(M)] [SIGNED | UNSIGNED | ZEROFILL]
  Eine kleine Ganzzahl im Wertebereich von -128 bis 127, falls vorzeichenbehaftet definiert. Der vorzeichenlose Wertebereich liegt von 0 bis 255. INT1 ist ein Synonym für TINYINT. BOOL und BOOLEAN sind Synonyme für TINYINT(1).

- BOOL, BOOLEAN
  Ein Wert von Null wird als falsch angesehen. Nicht-Null-Werte werden als wahr betrachtet.

- INT(EGER)[(M)] [SIGNED | UNSIGNED | ZEROFILL]
  Wenn UNSIGNED markiert ist, reicht sie von 0 bis 4.294.967.295, andernfalls ist ihr Bereich von -2.147.483.648 bis 2.147.483.647 (SIGNED ist die Standardeinstellung).

Datenbanken

String Datentypen

- Strings sind Zeichenfolgen und werden in Anführungszeichen gesetzt.
- Zeichenfolgen können entweder in einfache oder doppelte Anführungszeichen eingeschlossen werden (dasselbe Zeichen muss sowohl zum Öffnen als auch zum Schließen der Zeichenfolge verwendet werden).
- Das \ (Backslash-Zeichen) wird verwendet, um Zeichen zu entwerten ,z.B. 'MariaDB\'s new features'.
- [NATIONAL] VARCHAR(M) [CHARACTER SET charset_name] [COLLATE collation_name]
  National – vordefiniertes Characterset; Collate/Collation – Vergleichsordnung

Datenbanken

Daten und Zeitstempel

- DATE
  Der unterstützte Bereich ist „1000-01-01“ bis „9999-12-31“. Je nach DBMS muss sich nicht exakt an dieses Format gehalten werden, z.B. MariaDB akzeptiert 12*04*86.

- TIME [(<microsecond precision>)]
  Der Bereich ist '-838:59:59.999999' bis '838:59:59.999999'. Die Mikrosekunden-Präzision kann zwischen 0 und 6 liegen.

- DATETIME [(microsecond precision)]
  Werte im Format 'JJJJ-MM-TT HH:MM:SS.ffffff'.

Datenbanken

Andere Datentypen

- AUTO_INCREMENT
  Das Attribut AUTO_INCREMENT kann verwendet werden, um eine eindeutige Identität für neue Zeilen zu generieren. Wenn Sie einen neuen Datensatz in die Tabelle einfügen (oder beim Hinzufügen eines AUTO_INCREMENT-Attributs mit der ALTER TABLE-Anweisung) und das AUTO_INCREMENT-Feld NULL oder DEFAULT ist (im Falle eines INSERT), wird der Wert automatisch erhöht.

- NULL
  NULL steht für einen unbekannten Wert. Es ist keine leere Zeichenfolge (standardmäßig) oder ein Nullwert. Dies sind alles gültige Werte und keine NULL-Werte. Beim Erstellen einer Tabelle, können Spalten mit den NULL- bzw. NOT NULL-Klauseln so angegeben werden, dass sie NULL-Werte akzeptieren oder nicht.

Datenbanken

Wertebereichsintegrität

- Bei der Definition von Attributen sind optional Wertebereichseinschränkungen und Vorgabewerte möglich, beispielsweise:
  - NULL/Nicht NULL: NULL-Wert ist (nicht) zulässig.
  - Eindeutigkeit (UNIQUE): Attributwert muss über alle Datensätze hinweg eindeutig sein.
  - Standard (DEFAULT): Vorgabe eines Standardattributwertes, welches überschrieben werden kann.
  - Zustandsprüfung (CHECK): Vorgabe einer Bedingung, z.B. Alter >= 18.

Datenbanken

Beispiele für die Sicherstellung der Wertebereichsintegrität

```
CREATE TABLE Personen
(
	Id int NOT NULL UNIQUE,
	Nachname varchar(255) NOT NULL,
	Vorname varchar(255) NOT NULL,
	PersonAlter int CHECK (PersonAlter >= 18),
	Stadt varchar(255) DEFAULT 'Stadtname',
	PRIMARY KEY(Id)
);
```

Datenbanken

Datenbankabfrage

- Der SQL-Abfragebefehl wird mithilfe des SELECT-Befehls formuliert und kann bis zu sechs Komponenten enthalten. Die grundlegende Syntax hat folgende Form:

```
SELECT [ALL | DISTINCT] {spalten | *}
FROM tabelle [alias] [tabelle [alias]] ...
[WHERE {bedingung | unterabfrage}]
[GROUP BY spalten [HAVING {bedingung | unterabfrage}]]
[ORDER BY spalten [ASC | DESC]...];
```

Datenbanken

Datenbankabfrage

- Der Syntax lässt sich wie folgt verstehen:

Klausel	Erläuterung
SELECT [DISTINCT]	Wähle die Werte aus der/den Spalte [mehrfache Datensätze nur einmal] ...
FROM	... aus der Tabelle bzw. den Tabellen ...
WHERE	... wobei die Bedingung(en) erfüllt sein soll(en) ...
GROUP BY	... und gruppiere die Ausgabe von allen Zeilen mit gleichem Attributwert zu einer einzigen ...
HAVING	... wobei darin folgende zusätzliche Bedingung(en) gelten müssen/muss ...
ORDER BY [ASC/DESC]	... und sortiere nach den Spalten [auf- bzw. absteigend].

Datenbanken

Datenbankabfrage

- Selektion aller Datensätze der Relation Schüler?

Schüler				
S.-Nr.	Name	Vorname	Klasse	Klassen-lehrer
1	Markus	Klaus	11a	Müller
2	Lackmeier	Torben	12a	Heinrichs
3	Schumacher	Heinrich	11a	Müller
4	Dietrich	Conrad	11b	Winter
5	Jürgens	Sebastian	12a	Heinrichs

Datenbanken

Datenbankabfrage

- Selektion aller Datensätze der Relation Schüler?

```
SELECT *
FROM Schüler;
```

Schüler				
S.-Nr.	Name	Vorname	Klasse	Klassen-lehrer
1	Markus	Klaus	11a	Müller
2	Lackmeier	Torben	12a	Heinrichs
3	Schumacher	Heinrich	11a	Müller
4	Dietrich	Conrad	11b	Winter
5	Jürgens	Sebastian	12a	Heinrichs

Datenbanken

Datenbankabfrage

- Selektion aller Datensätze der Relation Schüler welche, bei denen Müller Klassenlehrer ist?

Schüler				
S.-Nr.	Name	Vorname	Klasse	Klassen-lehrer
1	Markus	Klaus	11a	Müller
2	Lackmeier	Torben	12a	Heinrichs
3	Schumacher	Heinrich	11a	Müller
4	Dietrich	Conrad	11b	Winter
5	Jürgens	Sebastian	12a	Heinrichs

Datenbanken

Datenbankabfrage

- Selektion aller Datensätze der Relation Schüler welche, bei denen Müller Klassenlehrer ist?

```
SELECT *
FROM Schüler
WHERE Klassenlehrer = 'Müller'
```

Schüler				
S.-Nr.	Name	Vorname	Klasse	Klassen-lehrer
1	Markus	Klaus	11a	Müller
2	Lackmeier	Torben	12a	Heinrichs
3	Schumacher	Heinrich	11a	Müller
4	Dietrich	Conrad	11b	Winter
5	Jürgens	Sebastian	12a	Heinrichs

Datenbanken

Datenbankabfrage

- Selektion der Spalten Name, Vorname und Klasse der Relation Schüler welche, bei denen Heinrichs Klassenlehrer ist?

Schüler				
S.-Nr.	Name	Vorname	Klasse	Klassen-lehrer
1	Markus	Klaus	11a	Müller
2	Lackmeier	Torben	12a	Heinrichs
3	Schumacher	Heinrich	11a	Müller
4	Dietrich	Conrad	11b	Winter
5	Jürgens	Sebastian	12a	Heinrichs

Datenbanken

Datenbankabfrage

- Selektion der Spalten Name, Vorname und Klasse der Relation Schüler welche, bei denen Heinrichs Klassenlehrer ist?

```
SELECT Name, Vorname, Klasse FROM Schüler WHERE Klassenlehrer = 'Heinrichs';
```

Schüler				
S.-Nr.	Name	Vorname	Klasse	Klassen-lehrer
1	Markus	Klaus	11a	Müller
2	Lackmeier	Torben	12a	Heinrichs
3	Schumacher	Heinrich	11a	Müller
4	Dietrich	Conrad	11b	Winter
5	Jürgens	Sebastian	12a	Heinrichs

Datenbanken

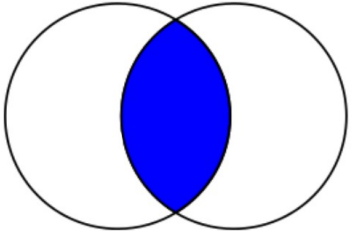
SQL Joins

- Ein SQL-Join (deutsch: Verbund) bildet aus den Datensätzen zweier Tabellen einer relationalen Datenbank eine Ergebnistabelle, deren Datensätze Attribute beider Tabellen entsprechend einer angegebenen Verbundbedingung enthält.
- Vier grundlegende Arten von Joins:
  - Inner Join
  - Left Join
  - Right Join
  - Full Join

Datenbanken

Inner Join

- Der INNER JOIN-Befehl gibt Zeilen zurück, die übereinstimmende Werte in beiden Tabellen haben.

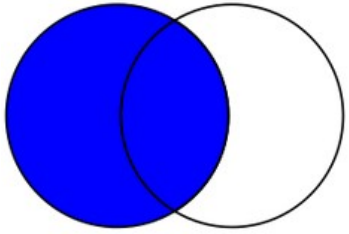


```
SELECT <fields>
FROM TableA A
INNER JOIN TableB B
ON A.key = B.key
```

Datenbanken

Left Join

- Der LEFT JOIN-Befehl gibt alle Zeilen aus der linken Tabelle und die übereinstimmenden Zeilen aus der rechten Tabelle zurück. Das Ergebnis ist NULL von der rechten Seite, wenn es keine Übereinstimmung gibt.

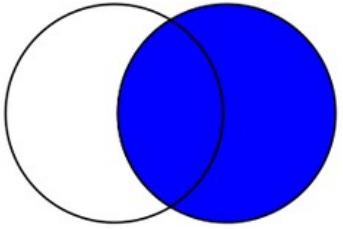


```
SELECT <fields>
FROM TableA A
LEFT JOIN TableB B
ON A.key = B.key
```

Datenbanken

Right Join

- Der RIGHT JOIN-Befehl gibt alle Zeilen aus der rechten Tabelle und die übereinstimmenden Datensätze aus der linken Tabelle zurück. Das Ergebnis ist NULL von der linken Seite, wenn es keine Übereinstimmung gibt.

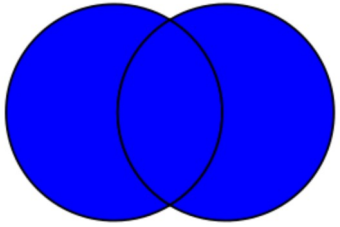


```
SELECT <fields>
FROM TableA A
RIGHT JOIN TableB B
ON A.key = B.key
```

Datenbanken

Full Join

- Das Schlüsselwort FULL JOIN gibt alle Datensätze zurück, wenn es eine Übereinstimmung in linken (Tabelle1) oder rechten (Tabelle2) Tabellendatensätzen gibt.



```
SELECT <fields>
FROM TableA A
FULL JOIN TableB B
ON A.key = B.key
```