

Informations- arkitektur och databasutveckling

Mikael Olsson





Socrative

<https://www.socrative.com/>

- Frågehanterare
 - Logga in som student
 - Ange rum "Emmio"
 - Få upp en vänta-skärm



Waiting for the next activity to begin...



Objekttyper

- Självständiga objekt (strong entities)
 - Oberoende av andra objekt
 - Egen lagringsnyckel
- Beroende objekt (weak entities)
 - Ägs av ett eller flera överordnade objekt
 - Kan t ex ha sammansatt nyckel av ägarens pk + egen del
 - Ex: telnr registreras sällan utan ägare



Objekttyper




FAKTURA

- Existerar en fakturarad utan en faktura?
- Vilken objekttyp?

Fakturanr		5/2011	Fakturaemottagare	
Datum		2011-02-24	Möbblgruvan	
Orderdatum		-	Brunnsgråtan 10	
Referens		Niklas Nilsson	776 35 Hedemora	
Betalningsvillkor		20 dagar netto		
Förfallodatum		2011-03-16		
Dröjsmålsränta		10 %		

Benämning	Summa
Webbutveckling del 1, se specifikation.	9 183 kr
Iframe till Faludäck, 1 timme	650 kr
Illustration omringning	2 000 kr
Illustration Maskot	2 000 kr

Sammanfatt nyckel



<u>FakturalID</u>	<u>RadID</u>	Benämning	Pris
1	1	1 <u>Webbutveckling del 1, se specifikation.</u>	9183
1	2	2 <u>Iframe till Faludäck, 1 timme</u>	650



Server vs Workbench

- Client / server
- Browser / web server
- Var finns servern?



Numeriska datatyper, exempel

Namn	Storlek signed	Storlek unsigned
INT	-2147483648 till 2147483647	0 till 4294967295
TINYINT	-128 to 127	0 till 255
SMALLINT	-32768 till 32767	0 till 65535

- Vad är skillnaden mellan *signed* och *unsigned*?
- Varför ska man välja en så anpassad typ till ens behov som möjligt?



Datum-datatyper, exempel

Namn	Format
DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIME	HH:MM:SS



Sträng-datatyper, exempel

Namn	Storlek
VARCHAR	1 till 255 tecken
TEXT	max 65535 tecken
ENUM	Lista

<https://www.tutorialspoint.com/mysql/mysql-data-types.htm>



Tabeller

Diagram illustrating database tables and their relationships:

Primärnyckel (Primary Key): Points to KundID in the Kund table.

Kolumnnamn (Column Name) / **Egenskapsnamn** (Property Name): Points to Namn in the Kund table.

Tabell (Table): Points to the Kund table.

Rader (Rows) / **Poster** (Records): Points to the data rows in the Kund table.

Kolumner (Columns) / **Egenskaper** (Properties): Points to the columns in the Kund table.

Främmande nyckel (Foreign Key): Points to the DistriktID column in the Kund table, which references the DistriktID primary key in the Distrikt table.

Kund		
<u>KundID</u>	<u>Namn</u>	Distrikt
1	Acme AB	2
2	BBB & Co	2

Distrikt	
<u>DistriktID</u>	Namn
1	Södra
2	Västra
3	Norra



Mer om nycklar

”The word key is one of the most overworked in the entire database field”

Primary key.....	Identifierar en rad unikt
Candidate key.....	Kolumner som kan fungera som pk
Secondary key.....	Kandidatnycklar som inte är pk
Surrogate key.....	Extra kolumn (autonumber) för pk
Alternate key.....	De kandidatnycklar som inte valdes till pk
Search key.....	Ett index avsett för att underlätta sökning
Foreign key.....	Primärnyckel från en annan tabell
Index key.....	Index på ett eller flera kolumner
Parent key.....	Pk på första sidan
Super key.....	En överbestämd pk
Composite key.....	Nyckel som består av flera kolumner
Ordering key.....	Ett index för att underlätta sorterad visning



Primärnyckel

- Huvudsyfte är att märka en post unikt
 - Består av ett eller flera fält
 - Väljs bland kandidatnycklar
 - Bör väljas stabil. Svårt (=dyrt) att ändra.
 - Undvik talande nycklar som artnr, persnr, namn osv
 - Undvik stora pk – ger stora index och dålig prestanda
 - Autonummerade nycklar är bra
 - Numeriska nycklar ger bäst prestanda.



Charset

- Tabell som representerar tecken
- Ju fler bitar ett tecken tar upp, desto fler tecken kan uppsättningen ha, men desto mer plats tar varje tecken upp också.
 - ASCII, ANSI
 - Olika ISO-uppsättningar
 - UTF8, UTF16, UTF32



Charset

- UTF8MB4
 - MySQLs UTF8 sparar max 3 bytes per tecken.
 - Motsvarar “Basic Multilanguage Plane”
 - UTF8MB4 ger utökat stöd, t ex emojis.



Collation

- Regler för sortering
- Ex: utf8mb4_swedish_ci
 - Reglerna gäller för utf8mb4
 - Svenska sorteringsregler
 - ci = Case insensitive
- Man kan sätta standard charset och collation per server, databas och tabell.



Escapa namn

- Vissa ord är reserverade.
- I MySQL escapar man ord med ` , t ex:

```
SELECT * FROM `new_schema`
```



MySQL Workbench

- Modellverktyget



Lab MySQL Workbench

- Skapa modell för orderhanteringssystemet



MySQL Workbench

- Skapa databas
- Skapa tabell / kolumner



Lab MySQL Workbench

- Implementera era modeller



Lab MySQL Workbench

- Implementera era modeller
- Lägg till ett par rader i varje tabell

Lunch!





Manipulera data

- CRUD
 - Create
 - Read
 - Update
 - Delete



Manipulera data

- SQL - Structured Query Language
 - INSERT INTO
 - SELECT
 - UPDATE
 - DELETE
 - -- Kommentar
 - /* Också kommentar */



Manipulera data

- `INSERT INTO [table] ([fields]) VALUES ([values]);`
- `INSERT INTO Person (`FirstName`, `LastName`)
VALUES ('Mikael', 'Olsson');`
 - Lägg märke till att ` inte är detsamma som '. Vilket används var?
 - Det finns varianter på `INSERT INTO`.



Manipulera data

- `SELECT [fields] FROM [table];`
- `SELECT FirstName, LastName FROM Person;`
- `SELECT * FROM Person; -- Obs, prestandakrävande!`



Manipulera data - Sortera

- `SELECT [fields] FROM [table] ORDER BY [field];`
- `SELECT FirstName, LastName
FROM Person
ORDER BY LastName DESC, FirstName;`



Manipulera data - Avgränsa

- `SELECT [fields] FROM [table] LIMIT [number],
[offset];`
- `SELECT FirstName, LastName
FROM Person
LIMIT 3, 0; -- Ger de tre första träffarna`
- MS SQL Server: `SELECT TOP 3 FROM [...]`



Manipulera specifik data - WHERE

- `SELECT [fields] FROM [table] WHERE [condition];`
- `SELECT FirstName, LastName FROM Person WHERE id = 23;`



Manipulera specifik data

- Operatorer
 - Relationsoperatorer
 - <, >, !=, >=, <=, =
 - Logiska operatorer
 - AND
 - OR
 - NOT
 - BETWEEN



Manipulera specifik data - WHERE

- `SELECT FirstName, LastName
FROM Person
WHERE age >= 23;`
- `SELECT FirstName, LastName
FROM Person
WHERE age BETWEEN 20 AND 65
OR NoOfCars > 5;`



Manipulera specifik data - LIKE

- ```
SELECT FirstName, LastName
FROM Person
WHERE LastName LIKE "Ols%";
```
- % = jokertecken



# Manipulera data

- `UPDATE [table] SET [field1] = '[value1]' WHERE [condition];`
- `UPDATE Person SET  
    FirstName = 'Mikael',  
    LastName = 'Olsson'  
WHERE id = 23;`
- Varför tittade vi på WHERE innan vi började uppdatera?





# Manipulera data

- `DELETE FROM [table] WHERE [condition];`
  - `DELETE FROM Person WHERE id = 23;`
- 
- Varför är WHERE viktigt här?



## Lab

- Allt i labben ska göras med SQL.
- Lägg till 10 personer i adressboken. Låt minst två personer ha namn som börjar på J.
- Ge alla personer adresser.
- Ge några personer 0 bilar, några 1 bil och några 2 bilar.
- Uppdatera 2 personers adress och telnr.
- Ta bort 2 personer.
- Visa alla personer som börjar på J.



# Git

Vad har ni för erfarenheter av versionshantering?

- Kopierad mapp
- final\_delivery2\_final\_revision4\_b\_copy.zip
- “Jag är inne i den här filen nu, ändra inte i den.”
- “Fasen, önskar att jag inte hade sparat den där filen.”
- “Jag kan hålla reda på vilka filer jag har ändrat i i huvudet.”



# Git

- Filer sparas i repo
- Git är ett distribuerat system, dvs alla användare har en egen kopia av repot
- Man kan jobba ihop med vem som helst som har samma repo, behöver inte vara en central server.



# Git

- Sätt upp din lokala kopia genom *clone*
- Skapa en ny “kopia av mappen” med *branch*
- Byt “mappkopia” med *checkout*
- Spara dina ändringar med *commit*
  - Tänk på att skriva ett vettigt commit-meddelande



# Git

- Hämta de senaste ändringarna med *pull*
- Skicka upp dina ändringar med *push*



## Lab

- Skapa ett github-konto
- Klona det här repot:  
<https://github.com/emmio-micke/syne17>
- Skapa en branch med ert förnamn. (Min branch skulle heta *micke*.)
  - Skippa å, ä, ö och liknande.
- Importera filen `<root>/sql files/classic_models.sql` i er Workbench
- Skapa mappen `<root>/addressbook`, spara er modell dit.
- Committa och pusha.
- Checka ut någon annans branch och titta på deras modell-fil.



# Förberedelser inför nästa tillfälle

- Gör en modell över era TV-tablåer i Workbench.
- Skapa tabellerna med hjälp av Workbench -> Database -> Forward Engineer.