Rättningsmall tenta den 25e oktober 2011

Uppgift 1

A) Null (Connolly/Begg, p.103) Represents a null value for an attribute that is currently unknown or is not applicable for this tuple. Its not a value an therfore <>0.
[at the moment unknowned value / not applicable value] and
[avoid trash-data inserts into db / not a value compared like zero (0)]

B) Stored procedure (Connolly/Begg, p.222) Stored procedures is a small program that can take parameters and be executed by users. Ypu can use them to do operations against several tables.
[parameters can be used] and
[to simplify operations/operations against several tables]

C) Primary Key (Connolly/Begg, p.101) The candidate key that is selected to identify tuples uniquely within the relation. It can not be NULL.
[Selected candidate key] and
[to identify each row uniquely]

D) Deadlock (Connolly/Begg, p.590) When two or more transactions are waiting for locks to be realesed that are held by each other.
[An ever lasting loop] and
[locks held by two transactions]

E) View (Connolly/Begg, p.105) Dynamic result of one or more relational operations operating on the base relation to produce another relation. A view is a virtual relation that does not necessarily exist in the database but can be produced upon request by a particular user at the time of request.
[change-independence -> applications]
[virtual table / based on relational operations] and [used as a security mechanism]

F) Transaction (Connolly/Begg, p.570) An action, or series of actions, carried out by a single user or application program, that reads or updates the contents of a database.
[betraktas som en helhet och kan inkludera många operationer] and
[kan bara som helhet lyckas fullt ut eller göras ogjord vid fel/egenskaper ACID]

Uppgift 2

se slides

Uppgift 3 (se 2.4 i kurslitteratur)

1. Data storage, retrieval and update. (Absolut viktigast)
2. A user-accessible catalog (viktig)
3. Transaction support
4. Concurrency control services (ja)
5. Recovery services (ja)
6. Authorization services
7. Support for data communication (ja)
8. Integrity services
9. Services to promote data independence
10. Utility services

Viktigt med argumentation som rättfärdigar valet av funktioner.

Uppgift 4

1. Musikportalen AB lagrar uppgifter om kunders beställningar i en enkel databas bestående av en tabell (se fig. nedan). Bolagets ägare använder databasen för att hålla reda på sina kunders beställningar. Analysera tabellen och diskutera designen i termer av *redundans*, *inkonsistens* och *normalformer/normalisering*. Ge även förslag på (designmässigt) bättre databaslösning i form av en slutlig E/R-modell i 3NF. För att få full poäng på frågan ska svaret vara väl strukturerat. (**12p**)

Beställningar

OrderNr	ArtikelNr	Artikeltyp	Vikt	Antal	Pris	LeverantörID	Leverantörsnamn	Leverantörsstatus
110133	EL200	JBL K-130	4	10	4000	L2200	Musikgrossisten AB	5
110298	H4100	Trumpall, svart	3	2	1500	L4403	Trumlagret HB	3
110133	EL300	JBL Diskanthorn	2	6	2900	L2200	Musikgrosisten AB	5
110298	H4101	Trumpall, krom	3	18	2000	L4403	Trumlagret HB	2
110499	G2200	Gitarr, akustisk	2	3	9500	L8806	Gitarrbyggarna HB	4
↓	↓	↓	↓	↓		↓		↓

Förutsättningar:

- tabellen har en primärnyckel {OrderNr, ArtikelNr}
- olika artiklar kan levereras av olika leverantörer. (Musikportalen AB har aldrig flera leverantörer av en och samma artikel)
- varje leverantör tilldelas en viss status (beroende på leveranssäkerhet)
- musikprodukterna har artikelnummer, pris, en viss vikt och kan finnas i olika typer eller utföranden

Inkonsistens: Motsägelsefulla data i vår databas, tex samma namn stavat på två olika sätt, eller två olika orter inlagda för samma företag. Hänger ihop med redundans. Får vi bort redundansen eliminerar vi samtidigt inkonsistensen.

I tabellexemplet kan vi se exemplet markerat med gult som är ett typexempel på inkonsistens som kan uppstå.

Redundans: Är när samma data förekommer på flera olika ställen. Innebär att risken för inkonsistenta data är stor eftersom man måste ändra på alla ställen den datan man ändrar finns. Redundans är förknippat med dålig databasdesign där normalisering ej beaktats tillräckligt.

I exemplet kan vi kika på Trumlagret HB som finns på två rader. Blir direkt problem om vi ska ändra namnet, vi måste då se till att ändra samtliga redundanta rader…

Normalisering: För att designa databasen på ett korrekt sätt behöver vi bryta isär tabellen Beställning för att slippa problem med inkonsistens, redundans. Till exempel är det problematiskt att uppdatera och ta bort data som finns lagrad på flera ställen (redundant). Glöms ett ställe bort löper vi risken att datan blir inkonsistent och ger två eller fler svar då frågor ska besvaras precist. Utan korrekt data blir databasen värdelös då kanske viktiga beslut ska fattas.

I samband med normalisering ska vi vi kika på normalformerna. Vi pratar då om funktionellt beroende (ett värde på varje b för varje a i en tabell), till exempel är ort beroende av postnummer, det finns då ett värde på orten för varje postnummer. Man säger att b är funktionellt beroende av a (ändras postnummret determinerar det värdet på orten som också måste kollas av så det stämmer då a ändras).

1a normalform handlar om att vi har en PK samt att det bara finns ett värde i varje fält/ruta. I tabellexemplet kan nog artikeltyp diskuteras. Troligen behöver den delas upp i två attribut. Ett attribut med artikelnamn och ett attribut med artikelbeskrivning.

(OrderNr, ArtikelNr, Artikelnamn, Artikeltyp, Vikt, Antal, Pris, LeveratörID, Leverantörsnamn, Leverantörsstatus)

2a normalform bygger vidare på att 1NF redan är uppfylld och att alla icke-nyckelattribut är funktionellt beroende av hela PK. Vi bryter ut Allt som beror på halva PK, dvs artikel och leverantörsdata. Leverantörsdata bryts i sin tur ut.

(OrderNr, ArtikelNr, Antal)

(ArtikelNr, Artikelnamn, Artikeltyp, Vikt, Pris, LeverantörID)

(LeveratörID, Leverantörsnamn, Leverantörsstatus)

3e normalform bygger vidare på 2a normalform och eliminerar alla transitiva beroenden mellan icke nyckelattribut. Status förefaller beroende av

(LeveratörID, Leverantörsnamn)

(Leverantörsnamn, Leverantörsstatus)

Rätt slutlösning i 3NF:

(OrderNr, ArtikelNr, Antal)

(ArtikelNr, Artikelnamn, Artikeltyp, Vikt, Pris, LeverantörID)

(LeveratörID, Leverantörsnamn)

(Leverantörsnamn, Leverantörsstatus)

Uppgift 5

```
SELECT Företagsnamn, Stad, tel FROM [Kund(Customer)] ORDER BY
Företagsnamn;

SELECT * FROM [Kund(Customer)] WHERE Företagsnamn LIKE '%AB' ORDER BY
Företagsnamn;

SELECT Datum, OrderID, Leveranssätt FROM [Order(Order)] WHERE KundID =
(SELECT DISTINCT KundID FROM [Kund(Customer)] WHERE tel ='031-785411');

SELECT Företagsnamn, Stad, tel, Kontaktperson,Datum FROM [Order(order)],
[Kund(Customer)] WHERE [Order(Order)].KundID=[Kund(Customer)].KundID AND
SäljareID='AK3';

UPDATE [Kund(Customer)] SET KundID=1004 WHERE KundID='1001';

DELETE FROM [Order] WHERE KundID=1002;
```

a-d)

	Företagsnamn	Stad	tel
1	Musikaliteten	Halmstad	035-121243
2	Musikportalen AB	Göteborg	031-785411
3	Musikportalen AB	Stockholm	08-163241

	KundID	Företagsnamn	Stad	tel	Kontaktperson
1	1001	Musikportalen AB	Stockholm	08-163241	Nils Larsson
2	1003	Musikportalen AB	Göteborg	031-785411	Anna Anka

	Datum	OrderID	Leveranssätt
1	2011-05-29 00:00:00	110211	Hämtas

	Företagsnamn	Stad	tel	Kontaktperson	Datum
1	Musikaliteten	Halmstad	035-121243	Eva Larsson	2011-03-20 00:00:00

e) Det går inte pga referensintegriteten.

```
The UPDATE statement conflicted with the REFERENCE constraint
"FK_Order(order)_Kund(Customer)". The conflict occurred in database "Jesper_material",
table "dbo.Order(order)", column 'KundID'.
The statement has been terminated.
```

f) 1 row affected

Uppgift 6

se slides