Tutorial - Week 12

Exam Revision Questions

Note: points allocation may not match points allocation for a similar question in the exam.

Question 1 Normalisation

1A. The table shown below is part of an office inventory database. Identify the design problems and draw a revised table structure in 3rd Normal Form (3NF) that corrects those problems. For each step explicitly identify and discuss which normal form is violated.

(Key: PK = Bold FK = Italic PFK = Bold + Italic)

Inventory (**ItemID**, Description, Quan, Cost/Unit, **Dept**, Dept Name, Dept Head)

ItemID and **Dept** is the candidate key for this table.

The following functional dependencies hold:

Dept ⟶ DeptName, DeptHead

ItemID ⟶ Description, Cost/Unit

Quantity, Cost/Unit ⟶ InventoryValue

ItemID	Description	Dept	Dept Name	Dept Head	Quantity	Cost/Unit	Inventory Value
4011	1.4m Desk	MK	Marketing	Jane Thompson	5	200	1000
4020	Filing Cabinet	MK	Marketing	Jane Thompson	10	75	750
4005	Executive chair	MK	Marketing	Jane Thompson	5	100	500
4036	1.2m Desk	ENG	Engineering	Ahmad Rashere	7	200	1400

Table 1. The Inventory table

(10 marks)

1B. Given the following relation (Inventory), and its functional dependencies - is it possible to demonstrate Armstrong's Axioms of Reflexivity, Augmentation and Transitivity?

Inventory (**ItemID**, Description, Quantity, Cost/Unit, **Dept**, DeptName, DeptHead(

ItemID and **Dept** is the candidate key for this table.

The following functional dependencies hold:

Dept ⟶ DeptName, DeptHead

ItemID ⟶ Description, Cost/Unit

Quantity, Cost/Unit ⟶ InventoryValue

(6 marks)

1NF: All data is atomic – there are no repeating groups. However, we can identify a repeating group in the relation
Inventory (ItemID, Description, Cost/Unit, (Qty, Dept, DeptName, DeptHead))

Removing repeating groups:
Item (ItemID, Description, Cost/Unit)
Inventory (*ItemID*, Qty, Dept, DeptName, DeptHead)

2NF: The table is in 1NF and there are no partial functional dependencies.
Partial functional dependency is Dept ⟶ DeptName and DeptHead
Item (ItemID, Description, Cost/Unit)
Department (Dept, DeptName, DeptHead)
Inventory (*ItemID*, Qty, *Dept*)

3NF: The relations are in 3NF because there are no transitive functional dependencies.

NB: Quan, Cost/Unit → Inventory Value
Because there is derived value between these columns (Quan × Cost/Unit = Inventory Value), this FD can be resolved by removing the redundant Inventory Value column.

3NF
Item (ItemID, Description, Cost/Unit)
Department (Dept, DeptName, DeptHead)
Inventory (*ItemID*, Qty, *Dept*)

Another approach

1NF: All data is atomic – there are no repeating groups. However, we can identify a repeating group in the relation
Inventory (Dept, DeptName, DeptHead (ItemID, Description, Qty, Cost/Unit)

Removing repeating groups:
Department (Dept, DeptName, DeptHead)
Inventory (ItemID, Description, Qty, Cost/Unit, *Dept*)

2NF: The table is in 1NF and there are no partial functional dependencies.
Partial dependency is ItemID ⟶ Description and ItemID ⟶ Cost/Unit

Item (ItemID, Description, Cost/Unit)
Department (Dept, DeptName, DeptHead)
Inventory (*ItemID*, Qty, *Dept*)

3NF: The above have no transitive dependencies so it is the solution.

3NF
Item (ItemID, Description, Cost/Unit)
Department (Dept, DeptName, DeptHead)
Inventory (*ItemID*, Qty, *Dept*)

1B

Reflexivity: Pick any two sets of attributes where one is a subset of the other.

E.g. {Dept} ⊆ {Dept, Qty} which then means

Dept, Qty ⟶ Dept

Augmentation: Take an existing FD and stick the same attribute to both sides. So from ItemID ⟶ Cost/Unit we can get:

ItemID, Qty ⟶ Cost/Unit, Qty

Transitivity: If we take (2) above together with the original 3rd dependency we can do:

ItemID, Qty ⟶ Cost/Unit, Qty AND Cost/Unit, Qty ⟶ Inventory Value gives us

ItemID, Qty ⟶ InventoryValue

Question 2 Transactions

A transaction is an indivisible unit of work. Explain what it means. Illustrate your explanation with writing a transaction that moves $500 from account number 324455 to account 783343.

Account table was created using the script below:

```
CREATE TABLE Account (AccNo int, AccountType char(10),
Balance decimal(10,2), PRIMARY KEY(AccNo));
```

To use a variable in SQL:

SET @name='John';

Assume that account balance is sufficient to withdraw $500.

Indivisible unit of work means that either the whole job gets done, or if an error occurs, the steps that have been completed need to be undone to return the database into the previous consistent state.

```
START TRANSACTION;

SET @amount = 500;

UPDATE Account SET balance = balance - @amount WHERE AccNo = 324455;
```

```
UPDATE Account SET balance = balance + @amount WHERE AccNo = 783343;
COMMIT;
```

Another acceptable approach (note this is not the full transaction):

```
SET @amount = 500;
SET @currBalance = (SELECT balance FROM Account WHERE AccNo = 324455);
SET @currBalance = @currBalance - @amount;
UPDATE Account SET balance = @currBalance WHERE AccNo = 324455;
```

Note, avoid hardcoding values in SQL transactions, use variables.

Question 3 Data Warehouse

3A. Transurban operates toll roads in Australia and the United States of America. Traffic data analysis helps determine road maintenance, RFID reader maintenance, gantry maintenance and capacity management to reduce inefficiencies in its tollway network.

Each part of the tollway is identified by a section (E3), and multiple sections (E1, E2, E3) make up a chunk, (C3) and multiple Chunks make up an entire roadway ("Eastlink"). Many different vehicles on toll roads every day. Each vehicle falls into one category: motorcycles, passenger vehicles, 4WD's, vans, trucks, prime movers, buses, trailers, recreational and miscellaneous machinery (e.g. tractors, cranes, street sweepers, back hoes).

Transurban's management wants to understand the vehicle trips on its road network. They need to understand the number of trips, trip length, trip duration,

Draw a *star schema* to support the design of this data warehouse, showing the attributes in each table. You do not need to select data types. Clearly display the legend for Primary Key, Foreign Key and Primary Foreign Key.

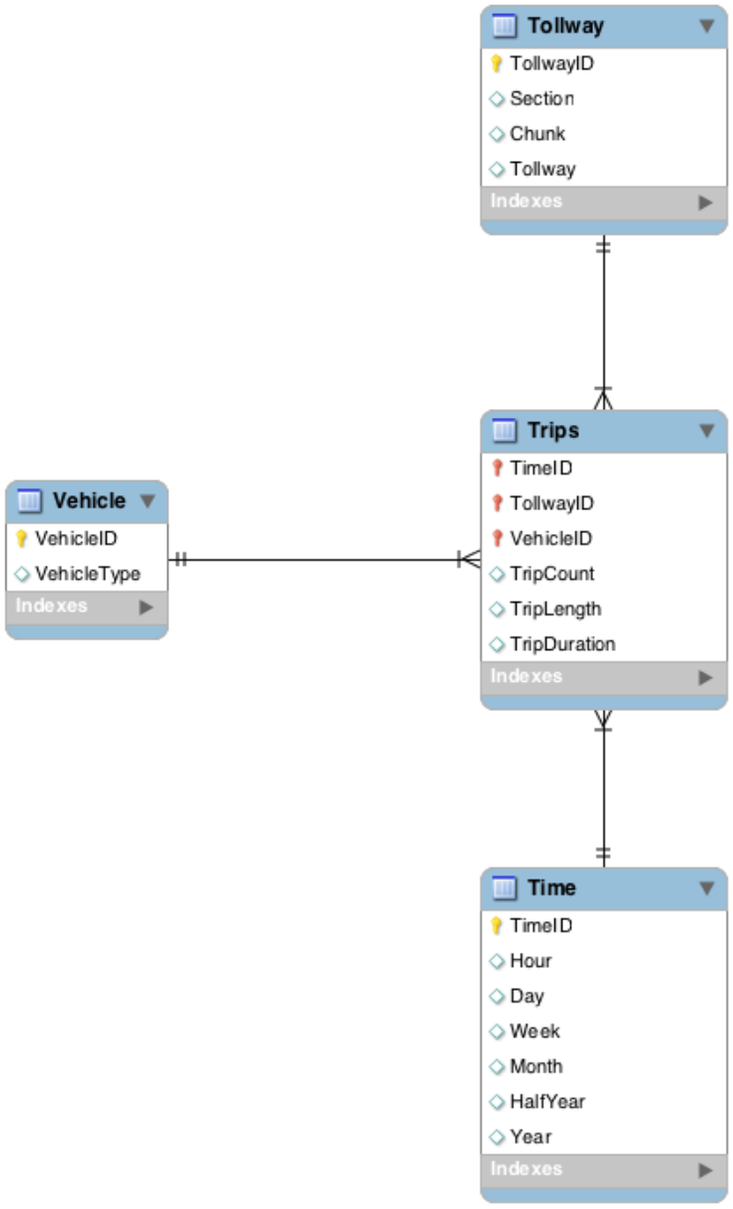
(8 marks)

See next page

3B. Why are star schemas preferred over relational database designs to support decision making?

(2 marks)

Star schemas support Data Warehouses which are organized around facts (business measures) and dimensions that help with managerial decision making. Star schemas are denormalised, making it faster to aggregate and query data. Faster aggregates and query data is required due to the large volume of data stored in data warehouses.


Tollway
TollwayID
Section
Chunk
Tollway
Indexes
Trips
TimeID
TollwayID
VehicleID
TripCount
TripLength
TripDuration
Indexes
Vehicle
VehicleID
VehicleType
Indexes
Time
TimeID
Hour
Day
Week
Month
HalfYear
Year
Indexes

Question 4 Transactions

A company has a table of products. Table name is Grocery

ItemID	Brand	Description	Weight	CurrentPrice	RRP	PurchPrice
19337	DairyWorks	Edam Sliced Cheese	500 g	7.50	7.50	3.50
19348	DairyWorks	Tasty Sliced Cheese	500 g	7.50	7.50	3.50
19375	Bega	Tasty Cheese Slices	500 g	7.00	9.50	3.00

At the beginning of each new sale period Current prices are updated to either be the same as RRP or determined by discounts applied to RRP.

Write a transaction that removes all previous week's discounts (makes Current price the same as RRP) and then deducts 50 cents from RRP to create a current price for Sliced Cheese by DairyWorks brand.

Solution

```
START TRANSACTION;
UPDATE Grocery SET CurrentPrice = RRP;
SET @discount = 0.50;
UPDATE Grocery SET CurrentPrice = RRP - @discount
   WHERE Brand = 'DairyWorks' AND Description LIKE '%Sliced Cheese';
COMMIT;
```