|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Practicum Final Exam – Odd Semester Year 2021/2022** | | | | | | |
| **Subject** | | | **ISYS6169001 – Database Systems** | | | Diagram, schematic  Description automatically generated |
| **Class** | **:** | **B001 / B101 / B601 / B701 / BE01 / BF01** | | **Start Date** | **: 12 January 2022** |
| **Lecturer** | **:** | **D4605 - Roy Kurniawan, S.Kom., M.M.**  **D4625 - Trisna Febriana, S.Kom., M.M.S.I**  **D6318 - Charles Bernando, S.Si., M.A., Ph.D** | | **Start Time** | **: 11:20** |
| **End Date** | **: 12 January 2022** |
| **End Time** | **: 13:20** |

**PERATURAN UJIAN:**

*Exam Regulations:*

* Mahasiswa tidak diperbolehkan berdiskusi dan/atau bekerja sama dengan peserta ujian lainnya

*Student is not allowed to discuss and/or work together with other exam participants*

* Mahasiswa tidak diperbolehkan untuk membuka dan menyalin dari **BUKU** atau **CATATAN**, **VIDEO** dari pengajar (recording kelas, VBL, Youtube, dsb) dan **REFERENSI** lainnya

*Student isn't allowed to open and copy from any resources such as notes, videos (class recording, VBL, Youtube, etc) and other references*

* Mahasiswa tidak diperbolehkan membuka dan menyalin jawaban dari internet (google, stackoverflow, dsb)

*Student isn't allowed to open and copy answer from the internet (google, stackoverflow, etc)*

* Asisten **BERHAK** memberi nilai 0 **(NOL)** bagi peserta ujian yang melakukan segala bentuk kecurangan

*Assistant is able to give 0 (ZERO) score for exam participant who does any cheating actions*

* Kumpulkan jawaban tepat pada waktunya, apabila terlambat mengumpulkan maka jawaban tidak akan dikoreksi dan nilai mahasiswa adalah 0

*Submit the answer on time, if not, then the answer will not be checked, and the students will receive 0 (ZERO)*

* Bila Anda tidak membaca peraturan ini, maka Anda dianggap telah membaca dan menyetujuinya

*If you have missed to read these regulations, so you are considered to have read and agreed on it*

Logo, company name

Description automatically generated

**SOFTWARE YANG DIGUNAKAN:**

*Software will be used:*

* SQL Server Developer 2019
* SQL Server Management Studio 18.9.1

**FILE YANG DIKUMPULKAN:**

*File must be collected:*

* SQL

**PERHATIAN!**

*Attention!*

* Bagi yang mengerjakan tidak sesuai dengan soal, maka akan diberikan nilai **NOL (0)**

*For those who do not work in accordance with the exam case will be marked as* ***ZERO (0)***

* Bagi yang mengerjakan tidak sesuai dengan software dan versi yang telah ditetapkan, maka akan tetap dikoreksi dengan software dan versi yang telah ditetapkan

*For those who do not work in accordance with the software and specific version will be corrected by the predefined software and version*

* Kompres semua jawaban yang akan diunggah. Pastikan format pengumpulan nama file dan ekstensi sesuai dengan format berikut: **[NIM]-[NAMA].zip**

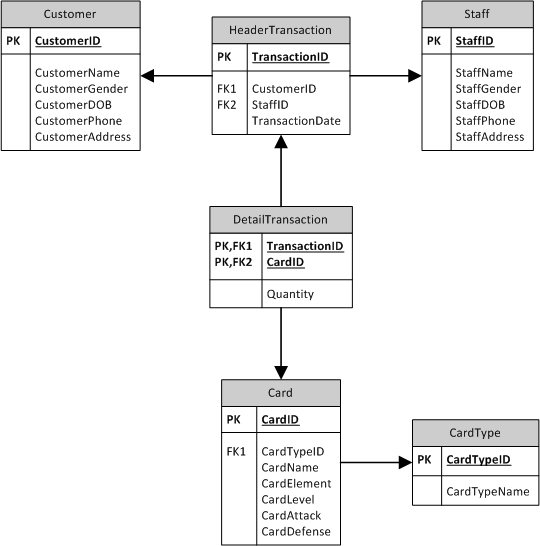
*Compress all file that will be uploaded. Make sure the format for collecting file name and extension according to the following format:* ***[NIM]-[NAME].zip***

**Tabel Relasional**

*Relational Table*

**Lugi Oh**

**Lugi Oh** is an online card game where players can collect many cards and battle with their own deck with another player. To get a card, the **player** must **buy** it from the **virtual store** and **verify** the payment with **payment** staff. The player as a **customer** can buy more than one kind of card in a transaction.

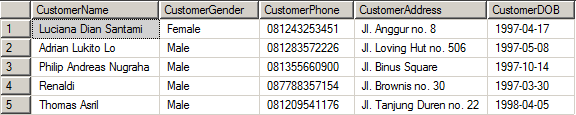


**Soal**

*Case*

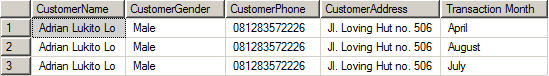
1. Display **CustomerName**, **CustomerGender**, **CustomerPhone**, **CustomerAddress**, and **CustomerDOB** data on Customer table for every Customer whose **name** **contains** letter **‘l’**.

(**like**)



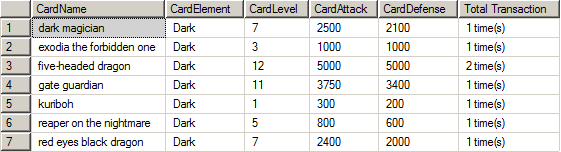
1. Display **CustomerName**, **CustomerGender**, **CustomerPhone**, **CustomerAddress**, and **Transaction Month** (obtained from the **name of the month** of **TransactionDate**).

(**datename, month**)

****

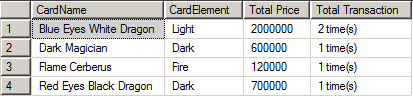
1. Display **CardName** (obtained from **CardName** in **lowercase format**), **CardElement**, **CardLevel**, **CardAttack**, **CardDefense**, and **Total Transaction** (obtained by adding **‘ times’** after the **total of transaction** that was already done) for every transaction which **CardElement** is ‘**Dark’**.

(**lower, cast, count, group by**)



1. Display **CardName**, **CardElement**, **Total Price** (obtained from the **sum of CardPrice**), and **Total Transaction** (obtained by adding **‘ times’** after the **total of transaction** that was already done) for every transaction that held **more than 8 months before 31 December 2017**. Then, **combine** it with **CardName**, **CardElement**, **Total Price** (obtained from the **sum of CardPrice**), and **Total Transaction** (obtained by adding **‘ times’** after the **total of transaction** that was already done) for every transaction which **CardPrice** is **more than 500000**.

(**sum, cast, count, datediff, month, group by**)



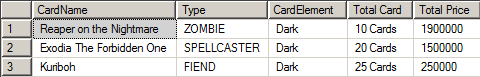
1. Display **CustomerName**, **CustomerGender**, and **CustomerDOB** (obtained from **CustomerDOB** in ‘**Mon dd, yyyy’** format) for every customer who **buys card on** **Friday**.

(**convert, in, day**)

D:\5.PNG

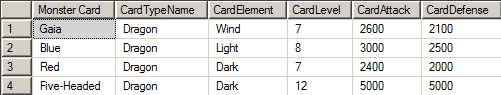
1. Display **CardName**, **Type** (obtained from **CardTypeName** in **uppercase format**), **CardElement**, **Total Card** (obtained by **adding ‘ Cards’ after Quantity**), and **Total Price** (obtained from **multiplying CardPrice and Quantity**) for every Card which **CardPrice** is **lower than the average price of all cards** and the **CardElement** is **‘Dark’**. Then, **sort** the data in **ascending** order based on **Quantity**.

(**upper, cast, alias subquery, avg, order by**)



1. Create a **view** named ‘**DragonDeck’** to display **Monster Card** (obtained from the **first word of CardName**), **CardTypeName**, **CardElement**, **CardLevel**, **CardAttack**, and **CardDefense** for every card which **CardTypeName** is **‘Dragon’**.

(**create view, substring, charindex**)



1. Create a **view** named **‘MayTransaction’** to display **CustomerName**, **CustomerPhone** (obtained by **replacing ‘8’** with **‘x’**), **StaffName**, **StaffPhone**, **TransactionDate**, and **Sold Card** (obtained from the **sum of quantity**) for every transaction that **occurs in May** and the **CustomerGender** is **‘Female’**.

(**create view, replace, sum, month, group by**)

D:\3.PNG

1. Add a **column** named **‘StaffSalary’** on Staff table with **integer** data type. After that, **add a constraint** to check that the **salary** must be **more than 100000**.

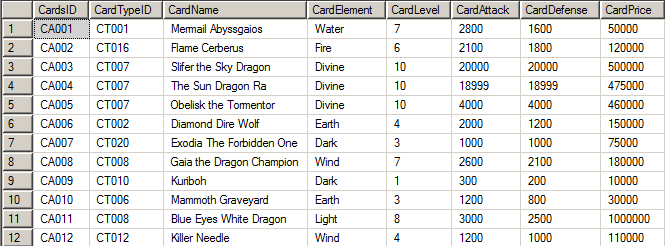
(**alter table, add, add constraint**)



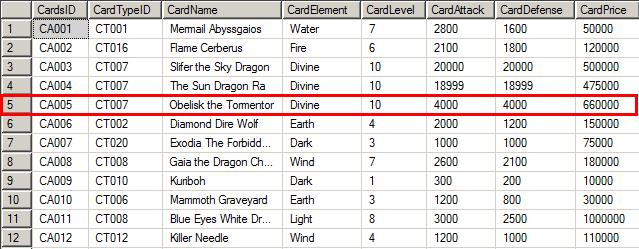
1. **Update** Cards data by **increasing** the **CardPrice** by **200000** on Cards table for every card that **CardTypeName** is **‘Divine-Beast’** and was sold with the **quantity more than 10**.

(**update**)

**Before update:**

****

**After update:**

**