Introduction

1. Purpose of Database Systems
2. View of Data
3. Data Models
4. Data Definition Language
5. Data Manipulation Language
6. Transaction Management
7. Storage Management
8. Database Administrator
9. Database Users
10. Overall System Structure

Other database models

A variety of other database models have been or are still used today.

Inverted file model

A database built with the inverted file structure is designed to facilitate fast full text searches. In this model, data content is indexed as a series of keys in a lookup table, with the values pointing to the location of the associated files. This structure can provide nearly instantaneous reporting in big data and analytics, for instance.

This model has been used by the ADABAS database management system of Software AG since 1970, and it is still supported today.

Flat model

The flat model is the earliest, simplest data model. It simply lists all the data in a single table, consisting of columns and rows. In order to access or manipulate the data, the computer has to read the entire flat file into memory, which makes this model inefficient for all but the smallest data sets.

Multidimensional model

This is a variation of the relational model designed to facilitate improved analytical processing. While the relational model is optimized for online transaction processing (OLTP), this model is designed for online analytical processing (OLAP).

Each cell in a dimensional database contains data about the dimensions tracked by the database. Visually, it's like a collection of cubes, rather than two-dimensional tables.

Semistructured model

In this model, the structural data usually contained in the database schema is embedded with the data itself. Here the distinction between data and schema is vague at best. This model is useful for describing systems, such as certain Web-based data sources, which we treat as databases but cannot constrain with a schema. It's also useful for describing interactions between databases that don't adhere to the same schema.

Context model

This model can incorporate elements from other database models as needed. It cobbles together elements from object-oriented, semistructured, and network models.

Associative model

This model divides all the data points based on whether they describe an entity or an association. In this model, an entity is anything that exists independently, whereas an association is something that only exists in relation to something else.

The associative model structures the data into two sets:

A set of items, each with a unique identifier, a name, and a type

A set of links, each with a unique identifier and the unique identifiers of a source, verb, and target. The stored fact has to do with the source, and each of the three identifiers may refer either to a link or an item.

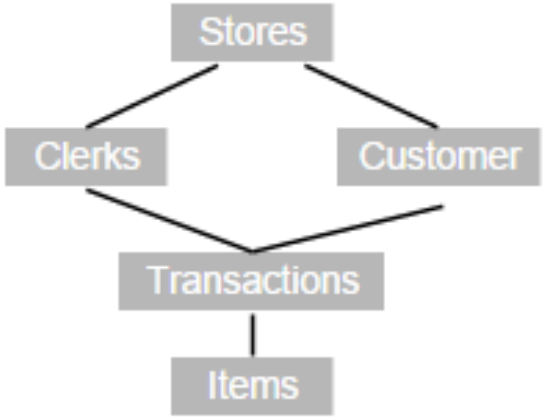
Other, less common database models include:
Semantic model, which includes information about how the stored data relates to the real world
XML database, which allows data to be specified and even stored in XML format
Named graph
Triplestore

NoSQL database models

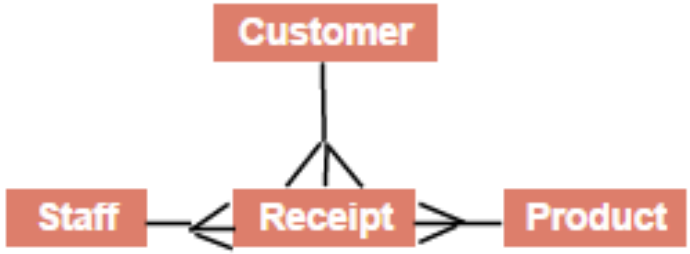
In addition to the object database model, other non-SQL models have emerged in contrast to the relational model:
The **graph database model**, which is even more flexible than a network model, allowing any node to connect with any other.
The **multivalue model**, which breaks from the relational model by allowing attributes to contain a list of data rather than a single data point.
The **document model**, which is designed for storing and managing documents or semi-structured data, rather than atomic data.

Databases on the Web

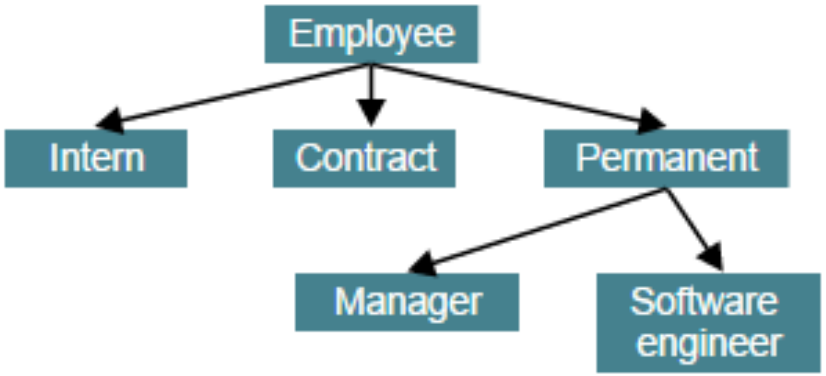
DBMS Models



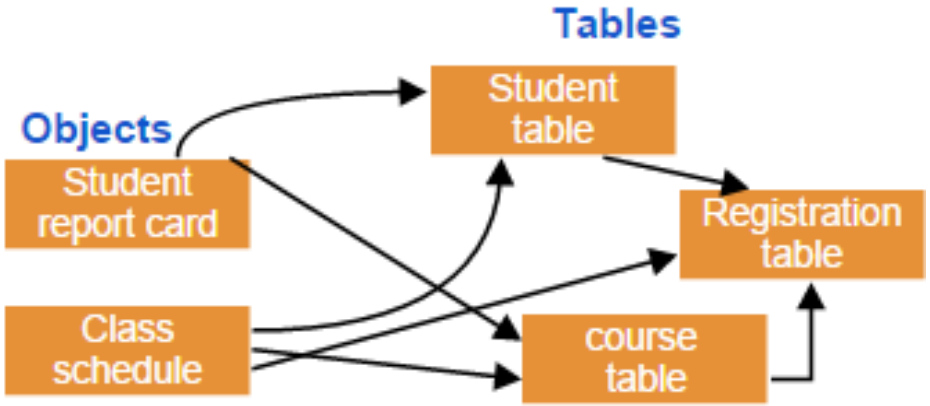
Network Database Model



Relational Database Model



Hierarchical Database Model

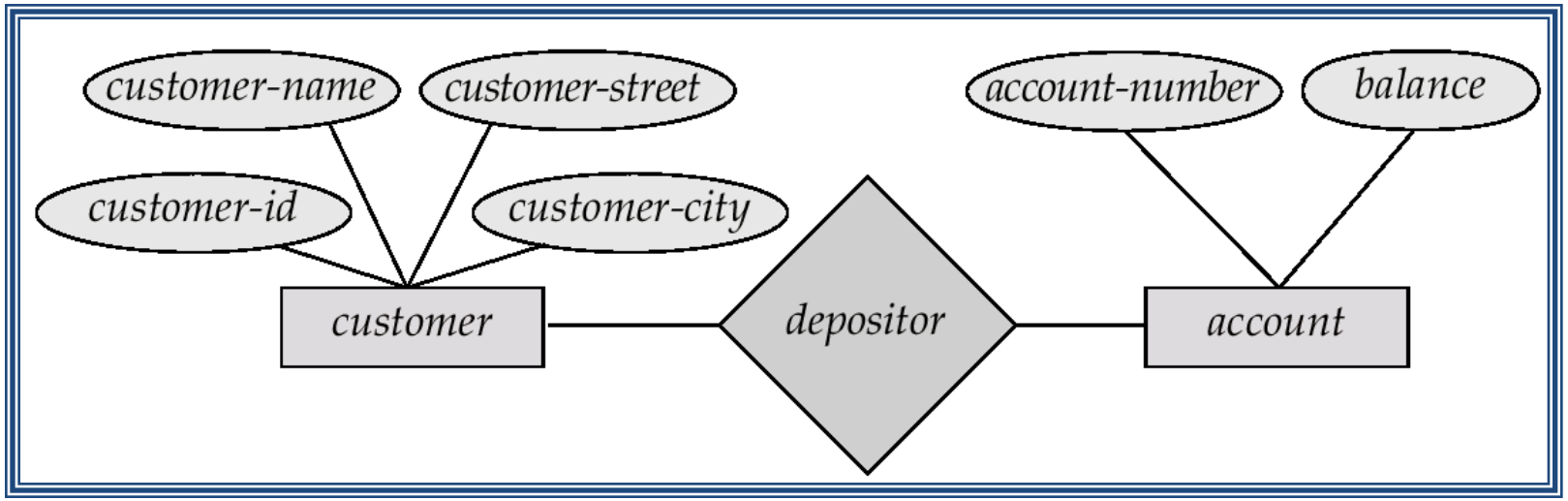


Object-oriented Database Model

BANERJEE; Dept of CSE;
partha.banerjee@juet.ac.in

Entity-Relationship Model

Example of schema in the entity-relationship model

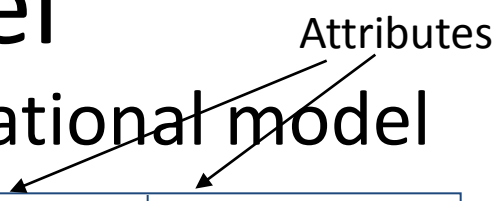


Entity Relationship Model (Cont.)

- E-R model of real world
  - Entities (objects)
    - E.g. customers, accounts, bank branch
  - Relationships between entities
    - E.g. Account A-101 is held by customer Johnson
    - Relationship set *depositor* associates customers with accounts
- Widely used for database design
  - Database design in E-R model usually converted to design in the relational model (coming up next) which is used for storage and processing

Relational Model

- Example of tabular data in the relational model



Customer-id	*customer-name*	*customer-street*	*customer-city*	*account-number*
192-83-7465	Johnson	Alma	Palo Alto	A-101
019-28-3746	Smith	North	Rye	A-215
192-83-7465	Johnson	Alma	Palo Alto	A-201
321-12-3123	Jones	Main	Harrison	A-217
019-28-3746	Smith	North	Rye	A-201

A Sample Relational Database

customer-id	*customer-name*	*customer-street*	*customer-city*
192-83-7465	Johnson	12 Alma St.	Palo Alto
019-28-3746	Smith	4 North St.	Rye
677-89-9011	Hayes	3 Main St.	Harrison
182-73-6091	Turner	123 Putnam Ave.	Stamford
321-12-3123	Jones	100 Main St.	Harrison
336-66-9999	Lindsay	175 Park Ave.	Pittsfield
019-28-3746	Smith	72 North St.	Rye

(a) The *customer* table

account-number	*balance*
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-217	750
A-222	700

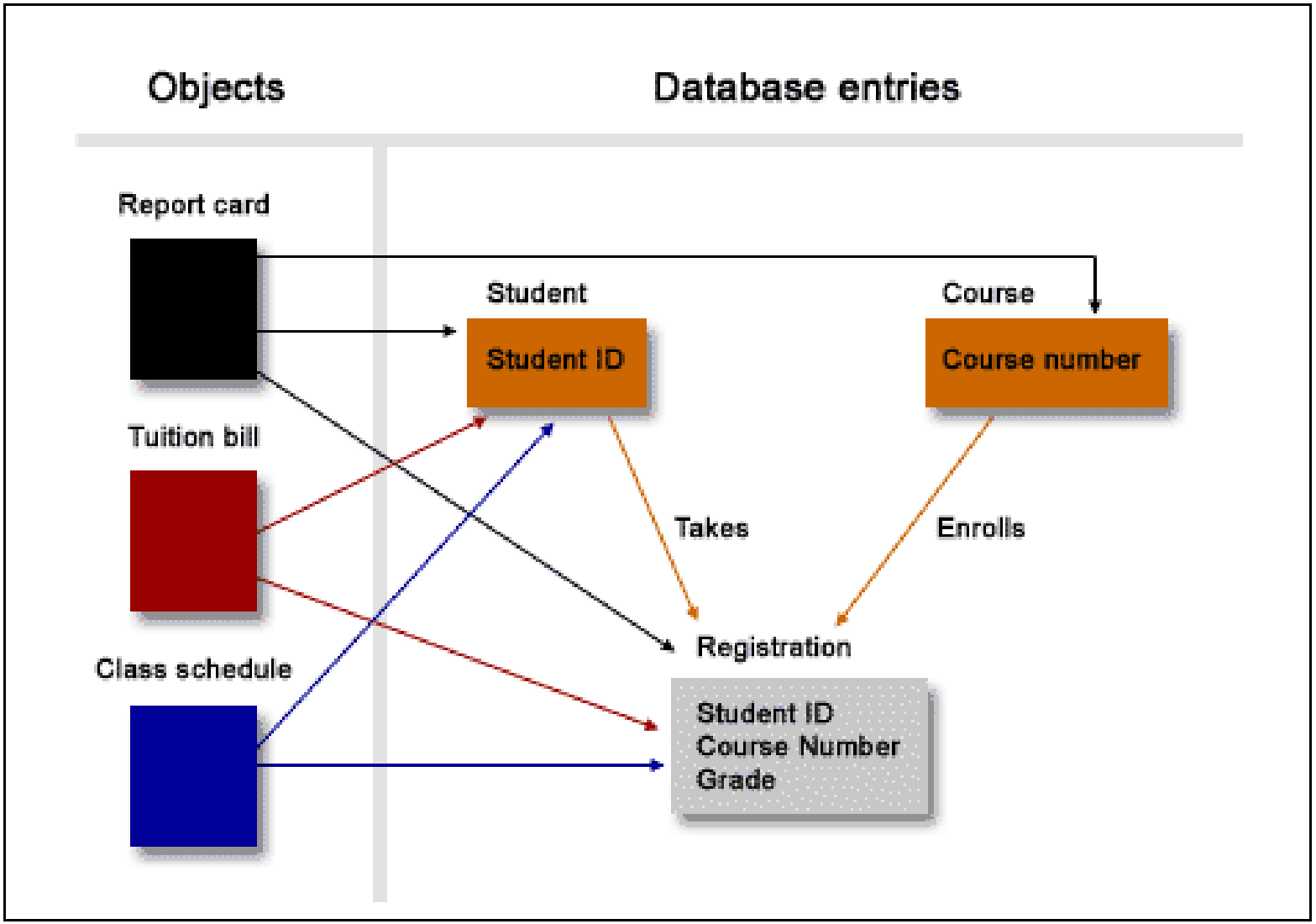
(b) The *account* table

customer-id	*account-number*
192-83-7465	A-101
192-83-7465	A-201
019-28-3746	A-215
677-89-9011	A-102
182-73-6091	A-305
321-12-3123	A-217
336-66-9999	A-222
019-28-3746	A-201

(c) The *depositor* table

Data Definition Language (DDL)

- Specification notation for defining the database schema
  - E.g.
    create table *account* (
    account-number **char**(10),
    balance **integer**)
- DDL compiler generates a set of tables stored in a *data dictionary*
- Data dictionary contains metadata (i.e., data about data)
  - database schema
  - Data *storage and definition* language
    - language in which the storage structure and access methods used by the database system are specified
    - Usually an extension of the data definition language


Objects
Database entries
Report card
Tuition bill
Class schedule
Student
Student ID
Course
Course number
Takes
Enrolls
Registration
Student ID
Course Number
Grade

Data Manipulation Language (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
  - DML also known as query language
- Two classes of languages
  - Procedural – user specifies what data is required and how to get those data
  - Nonprocedural – user specifies what data is required without specifying how to get those data
- SQL is the most widely used query language

DML

DML is abbreviation of **Data Manipulation Language**. It is used to retrieve, store, modify, delete, insert and update data in database.

Examples: SELECT, UPDATE, INSERT statements

DDL

DDL is abbreviation of **Data Definition Language**. It is used to create and modify the structure of database objects in database.

Examples: CREATE, ALTER, DROP statements

DCL

DCL is abbreviation of **Data Control Language**. It is used to create roles, permissions, and referential integrity as well it is used to control access to database by securing it.

Examples: GRANT, REVOKE statements

TCL

TCL is abbreviation of **Transactional Control Language**. It is used to manage different transactions occurring within a database.

Examples: COMMIT, ROLLBACK statements

A **data manipulation language (DML)** is a family of syntax elements similar to a computer programming language used for selecting, inserting, deleting and updatingdata in a database. Performing read-only queries of data is sometimes also considered a component of DML.

A data manipulation language (DML) is a family of computer languages including commands permitting users to manipulate data in a database. This manipulation involves inserting data into database tables, retrieving existing data, deleting data from existing tables and modifying existing data. DML is mostly incorporated in SQL databases.

Data Definition Language (DDL) is a standard for commands that **define** the different structures in a database. DDL statements create, modify, and remove database objects such as tables, indexes, and users. Common DDL statements are CREATE, ALTER, and DROP.

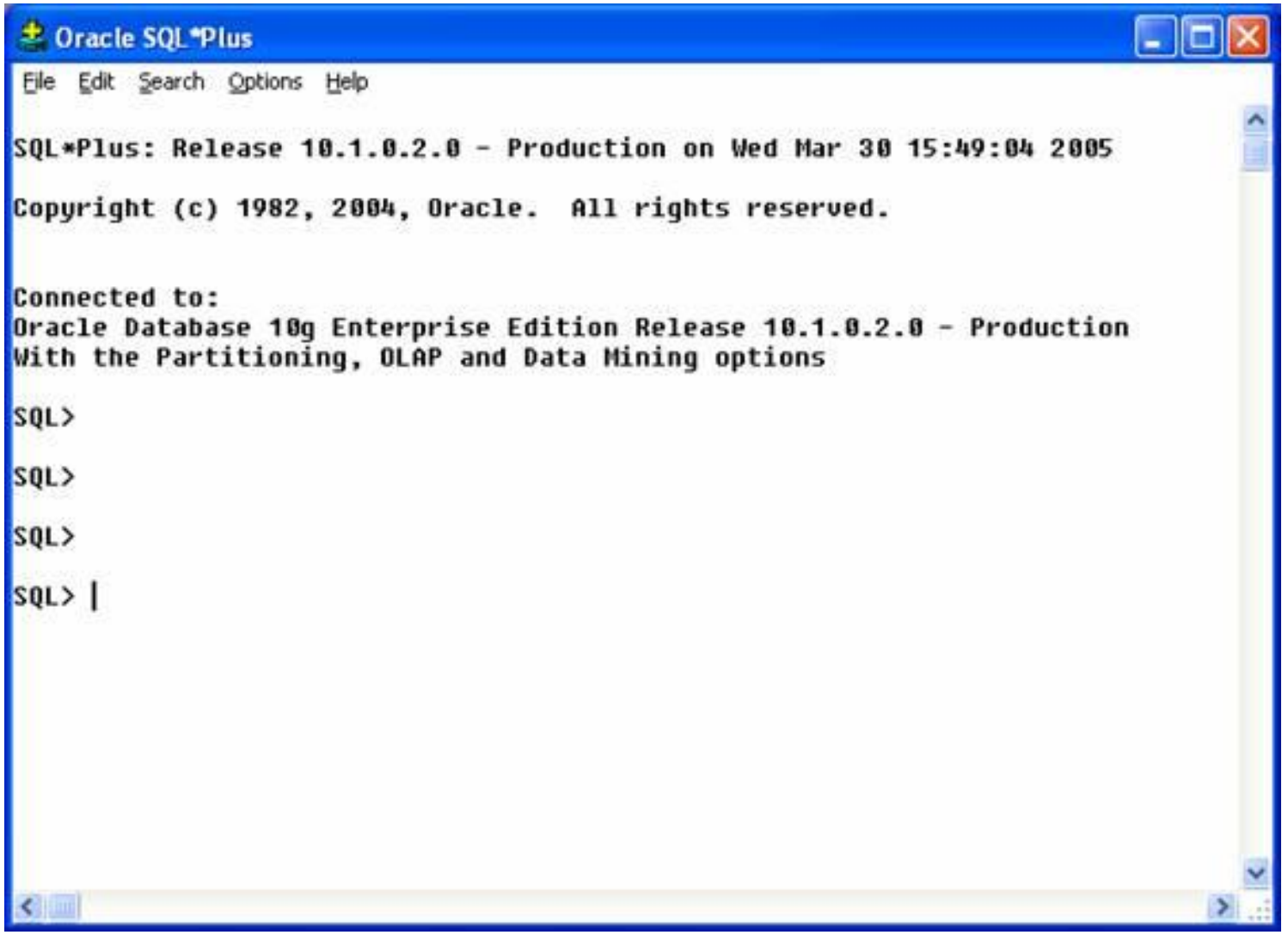
A **data control language** (DCL) is a syntax similar to a computer programming**language** used to **control** access to **data** stored in a database (Authorization). In particular, it is a component of Structured Query **Language** (SQL). Examples of DCL commands include: GRANT to allow specified users to perform specified tasks.

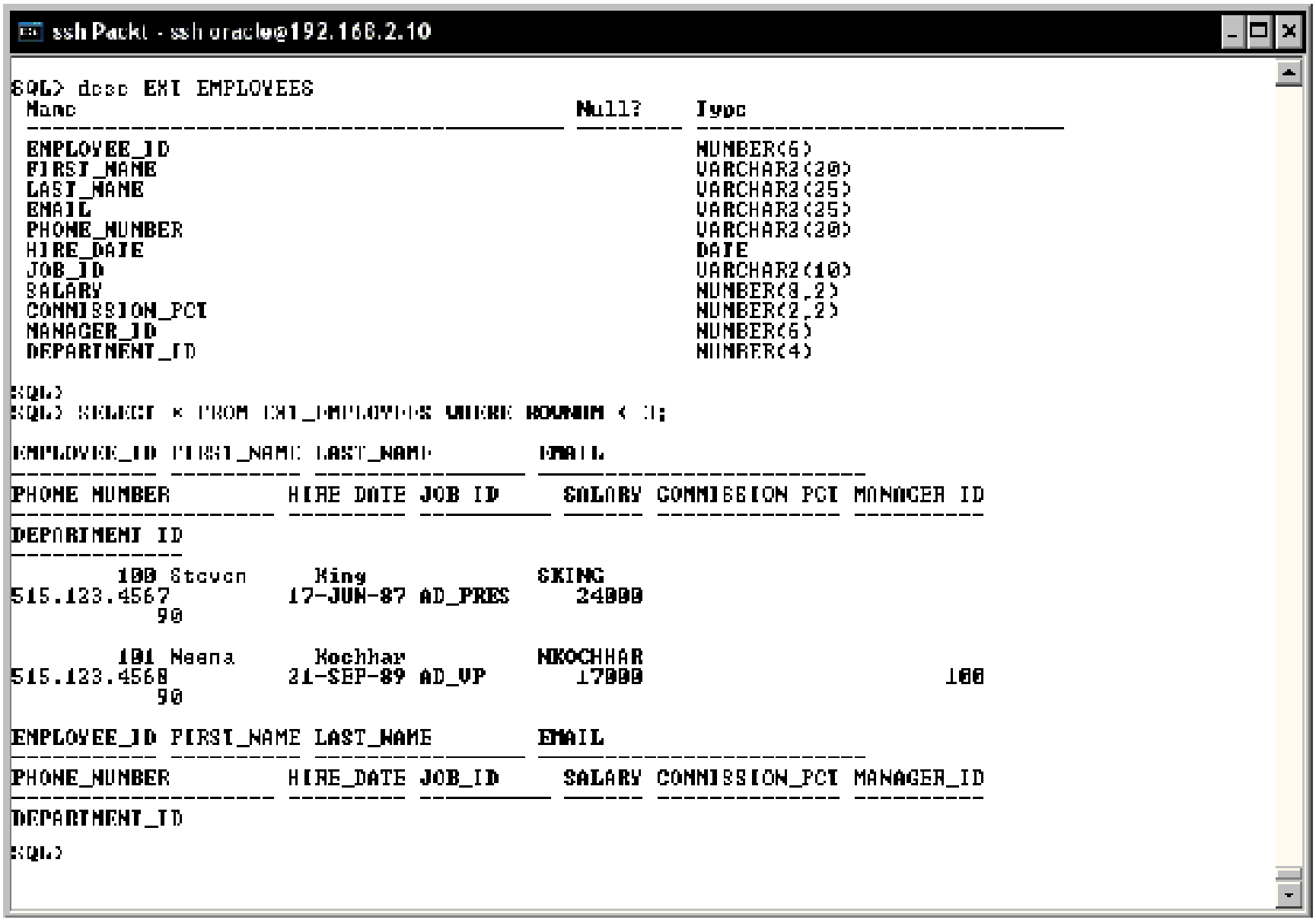
SQL

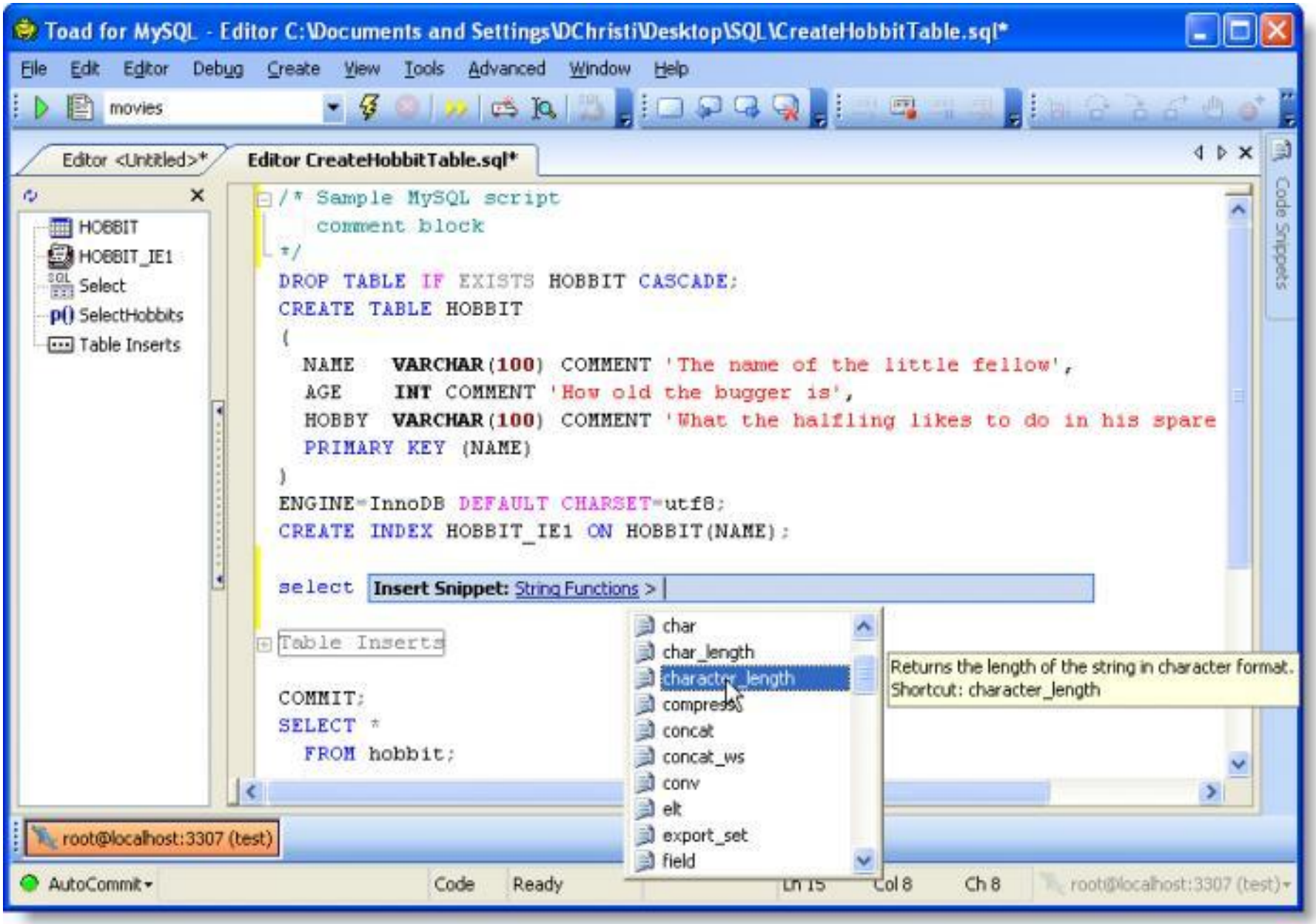
- SQL: widely used non-procedural language
  - E.g. find the name of the customer with customer-id 192-83-7465

    select *customer.customer-name*
    from *customer*
    where *customer.customer-id* = '192-83-7465'
  - E.g. find the balances of all accounts held by the customer with customer-id 192-83-7465

    select *account.balance*
    from *depositor*, *account*
    where *depositor.customer-id* = '192-83-7465' **and** *depositor.account-number = account.account-number*
- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interface (e.g. ODBC/JDBC) which allow SQL queries to be sent to a database


Oracle SQL*Plus
File Edit Search Options Help
SQL*Plus: Release 10.1.0.2.0 - Production on Wed Mar 30 15:49:04 2005
Copyright (c) 1982, 2004, Oracle. All rights reserved.
Connected to:
Oracle Database 10g Enterprise Edition Release 10.1.0.2.0 - Production
With the Partitioning, OLAP and Data Mining options
SQL>
SQL>
SQL>
SQL>


ssh Packt - ssh oracle@192.168.2.10
SQL> desc EXT_EMPLOYEES
Name Null? Type
EMPLOYEE_ID NUMBER(6)
FIRST_NAME VARCHAR2(20)
LAST_NAME VARCHAR2(25)
EMAIL VARCHAR2(25)
PHONE_NUMBER VARCHAR2(20)
HIRE_DATE DATE
JOB_ID VARCHAR2(10)
SALARY NUMBER(8,2)
COMMISSION_PCT NUMBER(2,2)
MANAGER_ID NUMBER(6)
DEPARTMENT_ID NUMBER(4)
SQL>
SQL> SELECT * FROM EXT_EMPLOYEES WHERE ROWNUM < 3;
EMPLOYEE_ID FIRST_NAME LAST_NAME EMAIL
PHONE_NUMBER HIRE_DATE JOB_ID SALARY COMMISSION_PCT MANAGER_ID
DEPARTMENT_ID
100 Steven King SKING
515.123.4567 17-JUN-87 AD_PRES 24000
90
101 Neena Kochhar NKOCHHAR
515.123.4568 21-SEP-89 AD_VP 17000 100
90
EMPLOYEE_ID FIRST_NAME LAST_NAME EMAIL
PHONE_NUMBER HIRE_DATE JOB_ID SALARY COMMISSION_PCT MANAGER_ID
DEPARTMENT_ID
SQL>


Toad for MySQL - Editor C:\Documents and Settings\DChristi\Desktop\SQL\CreateHobbitTable.sql*
File Edit Editor Debug Create View Tools Advanced Window Help
movies
Editor <Untitled>*
Editor CreateHobbitTable.sql*
HOBBIT
HOBBIT_IE1
Select
SelectHobbits
Table Inserts
/* Sample MySQL script
comment block
*/
DROP TABLE IF EXISTS HOBBIT CASCADE;
CREATE TABLE HOBBIT
(
NAME VARCHAR(100) COMMENT 'The name of the little fellow',
AGE INT COMMENT 'How old the bugger is',
HOBBY VARCHAR(100) COMMENT 'What the halfling likes to do in his spare
PRIMARY KEY (NAME)
)
ENGINE=InnoDB DEFAULT CHARSET=utf8;
CREATE INDEX HOBBIT_IE1 ON HOBBIT(NAME);
select
Insert Snippet: String Functions >
Table Inserts
COMMIT;
SELECT *
FROM hobbit;
char
char_length
character_length
compress
concat
concat_ws
conv
elt
export_set
field
Returns the length of the string in character format.
Shortcut: character_length
Code Snippets
root@localhost:3307 (test)
AutoCommit
Code
Ready
Ln 15
Col 8
Ch 8
root@localhost:3307 (test)

CONSTRAINTS:

There are three types of key constraints that are most common.

Primary Key constraint

Foreign Key constraint

Unique Key constraint

Many tables will have a primary key constraint and a table may only have one primary key constraint. The primary key is one or more columns (typically one) that uniquely identifies the row. Typical examples would be an employee id for a an employee table, a Stock Keeping Unit (SKU) for a product table, an order number for an orders table, and so on.

The DBMS will prevent a row from being inserted that results in any duplicate primary key— this is the 'constraint' part.

Foreign key constraints link two tables together. For instance, an Order table might have a customer id, that links the order to the customer id primary key in the customer table. The Order table's primary key is the order number, but one of its foreign keys is the customer id. Foreign keys are most often used to join two tables — from a foreign key in one table to a primary key in another table. The DBMS will prevent a row from being inserted if the foreign key column has no matching row with same key in the primary key column of the parent table. For example, you can't insert a row in the orders table with a customer number of 12345 unless there is a row in the customer table with customer id 12345.

This is the constraint the must be satisified. Similarly, you can't delete the customer row 12345 if there are any outstanding rows in the order table with that customer id. However, you can define the foreign key constraint to 'cascade' such that deleting the customer row will 'cascade' and delete the corresponding order rows. Finally, a unique key constraint is much like a primary key constraint, but just is not the 'knighted' primary key by which rows are primarily identified. For example, a customer id may be used as the primary key for the customer table, but you might also store the social security number and having duplicate social security numbers would be a red flag that someone may have stolen an identity. By making the SSN a unique primary key constraint, the application will prevent the new customer from being added to the database.